UNIVERSITÉ DE SHERBROOKE

Faculté de génie
Département de génie électrique et génie informatique

INCRUSTATION D'UN LOGO DANS UN FICHIER VIDÉO CODÉ AVEC LE STANDARD MPEG-2

Mémoire présenté au Département de génie électrique
en vue de l'obtention du grade de Maître en Génie (M.Sc.)

Membres du jury : Chon Tam Le Dinh, François Michaud et Jean Rouat

Patrick KEROULAS

Sherbrooke, (QC), Canada Juillet 2009

Your file Votre référence
ISBN: 978-0-494-53399-4
Our file Notre référence
ISBN: 978-0-494-53399-4

Résumé

Ce mémoire constitue l'aboutissement du projet de recherche de Patrick Keroulas et aborde la notion de compression vidéo, domaine en pleine ébullition avec la démocratisation de l'équipement vidéo et des réseaux de télécommunication. La question initiale est de savoir s'il est possible de modifier le contenu de l'image directement dans un flux binaire provenant d'une séquence vidéo compressée. Un tel dispositif permettrait d'ajouter des modifications en n'importe quel point d'un réseau en évitant le décodage et recodage du flux de données, ces deux processus étant très coûteux en termes de calcul.

Brièvement présentés dans la première partie, plusieurs travaux ont déjà proposé une gamme assez large de méthodes de filtrage, de débruitage, de redimensionnement de l'image, etc. Toutes les publications rencontrées à ce sujet se concentrent sur la transposition des traitements de l'image du domaine spatial vers le domaine fréquentiel. Il a été convenu de centrer la problématique sur une application potentiellement exploitable dans le domaine de la télédiffusion. Il s'agit d'incruster un logo ajustable en position et en opacité dans un fichier vidéo codé avec la norme *MPEG-2*, encore couramment utilisée. La transformée appliquée par cet algorithme de compression est la *DCT* (*Discrete Cosine Transform*). Un article publié en 1995 traitant de la composition vidéo en général est plus détaillé car il sert de base à cette étude. Certains outils proposés qui reposent sur la linéarité et l'orthogonalité de la transformée seront repris dans le cadre de ce projet, mais la démarche proposée pour résoudre les problèmes temporels est différente.

Ensuite, les éléments essentiels de la norme *MPEG-2* sont présentés pour en comprendre les mécanismes et également pour exposer la structure d'un fichier codé car, en pratique, ce serait la seule donnée accessible. Le quatrième chapitre de l'étude présente la solution technique mise en œuvre via un article soumis à *IEEE Transactions on Broadcasting*. C'est dans cette partie que toutes les subtilités liées au codage sont traitées : la structure en blocs de pixel, la prédiction spatiale, la compensation de mouvement au demi-pixel près, la nécessité

ou non de la quantification inverse. À la vue des résultats satisfaisants, la discussion finale porte sur la limite du système : le compromis entre son efficacité, ses degrés de liberté et le degré de décodage du flux.

Mots clés :

Vidéo, Image, *DCT*, *MPEG-2*, compression, codage.

Remerciements

Je remercie les personnes suivantes pour toute forme de contribution :

- Chon Tam LeDinh pour la direction, pour avoir valorisé l'intérêt du projet et pour ses encouragements,
- La Fondation de l'Université de Sherbrooke pour m'avoir accordé la Bourse de Soutient à la Recherche en Traitement d'Image,
- Les donateurs de la Fondation Force et de la Fondation de l'Université de Sherbrooke,
- Linda Simoncelli, Hélène Goudreau et Danielle Gagné pour avoir pu répondre à mes interrogations répétées,
- Kha Ledinh, François Rossignol et Jean François Trégouët pour toute forme de soutient technique et encouragement,
- Rafaël Correia, Isabelle Morin, Rachel Féron, Sophie Essiambre et mes parents pour les encouragements au quotidien.

Table des matières

Liste des figures

Liste des tableaux

Lexique

- Codec : terme employé pour désigner l'association codeur-décodeur.
- *GOP* : *Groupe Of Pictures* : séquence élémentaire d'images, commence toujours par une image I.
- Image I : pour le codage d'image intra.
- Image P : pour le codage temporel via prédiction.
- Image B : pour le codage temporel bidirectionnel.
- Bloc : échantillon image 8x8 pixels.
- Bloc *DCT* : échantillon image 8x8 pixels ayant subi une transformation *DCT*.
- Macrobloc : structure formée de 4 blocs de luminance et de plusieurs blocs de chrominances dont le nombre dépend du format de chrominance.
- Format de chrominance : type de sous-échantillonnage effectué sur les composantes de chrominance.
- *DCT* : *Discrete Cosine Transform*, Transformée en cosinus discret à deux dimensions.
- *I DCT* : Inverse *Discrete Cosine Transform*, Transformée en cosinus discret à deux dimensions inverse.
- *Q* : *Quantification*, opération de quantification sur un bloc.
- *I Q* : *Inverse Quantification*, opération de quantification inverse sur un bloc.
- *VLC* : *Variable Length Coding*, attribution de mot de longueur variable aux données du flux avant la transmission ; exemple: codage de type *Huffman*, *RLE*.
- *MC* : *Motion Compensation*, compensation de mouvement.
- *ME* : *Motion Estimation*, recherche du vecteur de mouvement d'un macrobloc.
- *MV : Motion Vector*, vecteur de mouvement d'un macrobloc pour une image Inter (P et B).
- RVB : Espace de couleur (Rouge, Vert, Bleu) d'une image pour l'affichage.
- YCrCb : Espace de couleur d'une image pour le codage. Y : Luminance, Cr = Y - R: Chrominance d'après le Rouge, Cb = Y – B : Chrominance d'après le Bleu.
- YUV : Autre nom utilisé pour désigner l'espace YCrCb.

1 INTRODUCTION

1.1 Contexte

Dans cette ère de l'information, la vidéo et l'image en général ont largement contribué à l'augmentation exponentielle de la quantité de données numériques du fait de leur démocratisation, notamment à travers Internet et autres réseaux de diffusion. Cependant, le débit limité des canaux de transmission impose la compression de ces données vidéo, trop volumineuses à l'état brut. De plus, il est toujours nécessaire d'optimiser les ressources matérielles et logicielles dont on dispose. Ainsi, les organismes *ISO* (*International Organization for Standardization*), *IEC* (*International Electrotechnical Commission*), et plus particulièrement le groupe *MPEG* (*Moving Picture Experts Group*), ont développé plusieurs générations de norme de codage vidéo (ou codec pour codeur-décodeur) que l'on retrouve sous les noms *MPEG-1, 2* ou *4*.

Du signal brut provenant d'un studio à l'image affichée sur l'écran pour un spectateur, le signal vidéo passe au travers d'une chaîne de transmission/traitement (modification des dimensions de l'image, correction de contraste ou de couleur, réduction d'un bruit parasite, etc.). Comme les données sont codées, nous n'avons pas accès à l'information élémentaire de l'image, le pixel. Toute modification nécessite traditionnellement de décoder au préalable le signal transmis et de le recoder ensuite. Ces processus sont relativement complexes et coûteux en temps de calcul, surtout au niveau du codeur. Il est alors intéressant de se demander s'il est possible d'appliquer les traitements d'images directement sur le signal codé.

1.2 Objectifs

Afin de cibler une application encore à l'étude dans l'industrie de la télédiffusion, il a été convenu avec le Directeur Chon Tam Le Dinh de réaliser l'incrustation d'une petite image fixe (le logo de l'Université de Sherbrooke) dans un fichier vidéo. Le programme permet à l'utilisateur de placer ce logo à n'importe quel endroit dans l'image et avec opacité ajustable. Le choix de la norme de compression s'est porté sur la norme *MPEG-2* car elle est encore très répandue, notamment dans la télévision numérique. Cette application serait tout à fait utile dans le cadre de la télédiffusion. Le lieu où l'image brute est générée et les lieux de traitement (studio de postproduction) et de diffusion sont séparés. Or, tous les éléments supplémentaires tels que les sous-titres, les graphiques et les logos ne sont pas ajoutés à la source mais bien en aval. De plus, il est prévu que l'application s'exécute en temps réel.

1.3 Méthodologie

La clé d'un algorithme de compression passe par la transposition des données dans un espace qui révèle leurs composantes principales par rapport à celles qui pourraient être supprimées. La norme *MPEG-2* utilise la transformée *DCT-II* (*Discrete Cosine Transform Type 2*), la même que pour la norme *JPEG*, employée pour le codage d'images fixes. Elle possède quelques propriétés mathématiques intéressantes qui rendent le domaine fréquentiel assez simple à manipuler. La première partie de ce mémoire présente une gamme assez large de possibilités de traitement dans le domaine *DCT*. L'état de l'art se poursuit par la description de la référence principale : (Chang et al., 1995). Ces auteurs posent tous les pré-requis nécessaires à la composition vidéo de plusieurs objets. Enfin une discussion est apportée sur les travaux les plus avancés en la matière. Il s'agit de (Panusopose et al., 2001) et (Roma et al., 2002)qui proposent une structure de transcodeur que l'on cherche à éviter dans ce projet.

L'objectif de la deuxième partie est de maîtriser le principe de fonctionnement de la norme de codage pour appréhender toutes les contraintes qu'elle impose (la structure en blocs de pixels, la prédiction spatiale, la compensation de mouvement au demi-pixel près, ...). En ce qui concerne la mise en œuvre logicielle, l'application générée à partir d'un code en *C* parcourt le

fichier *MPEG*-2 pour, d'une part, en extraire les informations utiles (information de localisation, paramètres de codage) et, d'autre part, localiser les données destinées à être remplacées. Le second objectif est donc d'exposer la structure hiérarchisée et complexe de l'information telle qu'elle se présente dans le flux de données.

C'est dans le quatrième chapitre que l'algorithme est expliqué via un article soumis et en cours d'évaluation à *IEEE Transactions on Broadcasting*. Le manuscrit commence par expliquer les outils de manipulation de bloc *DCT*, les intègre pour réaliser l'incrustation du logo dans une image fixe puis en mouvement. Une section est aussi destinée à apporter les subtilités techniques liées au codage.

Enfin, le dernier chapitre présente le protocole expérimental résumé par la **Error! Reference source not found.**. Le signal d'entrée du module d'incrustation dans le domaine *DCT* est un fichier généré par le codeur MPEG-2. Ce même flux de données codées sert également de référence pour évaluer la performance de l'algorithme. En effet, une fois décodées, les images de référence subissent l'incrustation du logo pixel par pixel, c'est-à-dire que l'opération est effectuée dans le domaine spatial. Ce même chapitre présente également l'environnement de développement de l'application supervisé par une interface graphique. Et surtout, des résultats supplémentaires viennent compléter l'étude exposée dans l'article, révélant mieux les paramètres qui influencent la qualité de l'image de sortie.

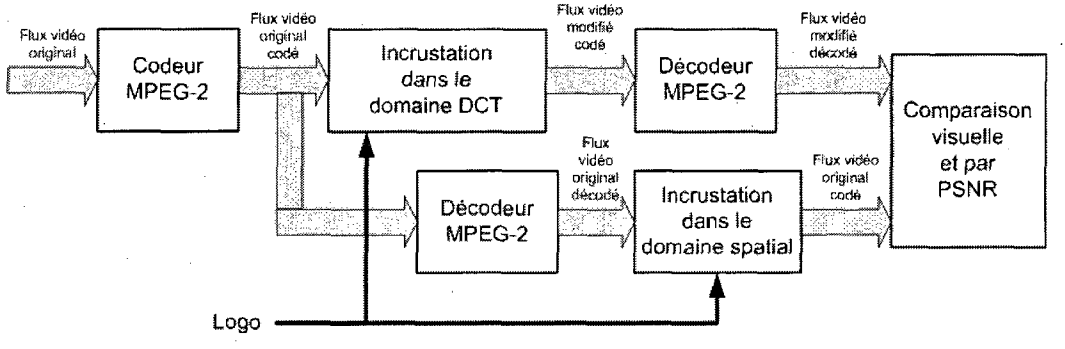


Figure 1-1 Protocole expérimental : comparer la technique d'incrustation d'un logo dans un flux vidéo dans le domaine *DCT* avec la technique dans le domaine spatial

1.4 Commentaires sur la contribution

L'algorithme constitue une mise en application d'outils établis par la principale étude antécédente sur la composition vidéo (Chang et al,. 1995). Cependant, la méthode mise en place dans le cadre de ce projet ne se contente pas de reprendre toute sa démarche. Elle prend clairement ses distances quand il s'agit de gérer les problèmes liés au codage temporel. Les résultats sont évalués sur la base d'un jugement a priori de la qualité subjective des images et d'un rapport signal sur bruit calculé par rapport aux images de référence. L'algorithme ne sera donc pas jugé sur la base d'une comparaison avec les résultats des études antérieures car leurs conditions d'expérimentation diffèrent. Leurs types de signaux à incruster ne sont pas les mêmes que celui prévu par les spécifications de ce projet, à savoir, un simple logo rectangulaire. Par exemple, (Chang et al,. 1995) utilisent une deuxième entrée vidéo sans mentionner la possibilité de transparence et (Roma et al., 2002) prennent un logo de forme arbitraire. De plus, ces auteurs n'explicitent pas toujours le profil d'utilisation de leur codec ni le format des images de chaque composante de couleur.

Pour finir, il faut considérer le problème du degré de décodage du fichier *MPEG*, c'est-à-dire le nombre d'opérations à effectuer pour obtenir les informations exploitables. On se rend vite compte que l'exigence initiale de non décodage semble impossible au sens strict, surtout si l'on souhaite gérer l'opacité du logo et sa position au pixel près. Le domaine *DCT* permet de manipuler simplement ces deux notions, mais encore faut-il accéder à ce domaine. Le prix à payer est un décodage partiel du flux de données. Le problème qui se pose est donc de vérifier que ces opérations de décodage ne rendent pas le processus plus exigent en terme de calcul que la méthode traditionnelle opérant sur les pixels. Quantifier et comparer la complexité de calcul permettrait de s'assurer du bien-fondé de procéder dans le domaine *DCT* plutôt que dans le domaine spatial. Cette justification a été fournie par des études statistiques (Merhav et al., 1996) et elle fait donc partie de nos hypothèses.

2 TRAVAUX ANTÉRIEURS DANS LE DOMAINE DCT

Les premières publications à propos du traitement d'image dans le domaine *DCT* datent d'une vingtaine d'années. Le but de ce chapitre est de présenter quelques exemples de transposition d'opérations qui s'effectuent habituellement dans le domaine spatial (sur l'image en tant que telle) vers le domaine de transformée, en exploitant les propriétés particulières de la *DCT*. Cette sélection est d'abord assez large et finit par cibler des applications directement liées au projet. Mais avant, il convient de donner quelques précisions sur cette notion de *DCT*.

Tout algorithme de codage transpose les données dans un autre domaine de telle sorte que les composantes principales ressortent par rapport à d'autres composantes moins significatives qui peuvent être supprimées. La transformation utilisée par le codec *MPEG-2* est la *Discrete Cosine Transform Type 2* ou *DCT –II*. Sa définition est détaillée en même temps que les spécifications de la norme de codage dans le chapitre 3. Appliquée à des segments de 8x8 pixels, elle nécessite normalement 1024 multiplications, mais (Cho et al., 1991) proposent une méthode d'optimisation pour atteindre 88 multiplications. Cette information n'est pas fondamentale dans le cadre de cette étude puisque les données du flux de données sont déjà des coefficients *DCT* et que le but du projet est d'éviter la transformée inverse (*IDCT*) ainsi que la *DCT* qui suit pour le recodage. Par contre, le logo présent sous forme de *bitmap,* ou matrice de pixels, doit être transposé dans le domaine *DCT* donc la question d'optimisation serait pertinente dans le cas d'une implémentation matérielle de l'algorithme.

2.1 Exemples de traitements dans le domaine *DCT*

2.1.1 Filtrage

Le filtrage des images dans le domaine spatial, c'est-à-dire la convolution des pixels avec un filtre, est facilement transposable dans le domaine fréquentiel en devenant une simple

multiplication dans le cas d'une transformée de Fourier discrète. Cependant, la *DCT* fait partie d'une autre famille : les transformées trigonométriques discrètes (*DTT*) pour lesquelles la transposition n'est pas aussi simple. (Chitprasert et al., 1990) réussissent à transposer la convolution vers le domaine *DCT* à la condition que le filtre ait une réponse fréquentielle réelle et paire. (Martucci et al., 1994) étendent la théorie à toutes les *DTT* et (Merhav et al., 1999) en profite pour créer un filtrage rapide en s'appuyant sur la combinaison des données dans les domaines *DCT* et *DST* (*Discrete Sine Transform*). Dans certains cas de filtrage d'images, (Changhoon, 2004) décomposent le problème en deux filtrages, un horizontal et un vertical. Cette supposition donne de bons résultats en termes d'efficacité et de complexité. Il faut porter attention au fait que tous ces travaux sont équivalents à une convolution spatiale pour laquelle les blocs de pixels sont étendus et/ou repliés. La rapidité du filtrage est assurée par la parcimonie des coefficients *DCT*. Cependant, le filtrage crée des discontinuités entre les blocs puisque ces éléments sont traités séparément les uns des autres.

2.1.2 Réduction des artéfacts de blocs

Ces artéfacts de blocs peuvent aussi apparaître tout simplement parce qu'à de fortes compressions, la quantification des coefficients *DCT* est trop grossière. Les approches fréquentielles ont très souvent été explorées, parfois même avec plus de réussite que par un filtrage spatial. Ces travaux ont en commun le fait de détecter la discontinuité entre un bloc et ses voisins en mesurant leur activité fréquentielle. Une réorganisation des coefficients *DCT* de plusieurs blocs permet de regrouper les coefficients d'une même sous bande de fréquence, ce qui facilitent l'analyse puis le traitement. On retrouve alors toutes les techniques de filtrage adaptatif. Certains auteurs (Triantafyllidis et al., 2002) se servent de l'erreur de quantification estimée comme facteur à minimiser; d'autres exploitent l'effet de masquage du système visuel humain pour traiter différemment les zones unies et les zones détaillées (Chen et al., 2001). (Paek et al., 1998) projettent la *DCT* d'un bloc à N points et un bloc voisin sur une *DCT* à $2N$ points pour obtenir l'activité fréquentielle globale de la région. Si cette dernière diffère trop de l'activité locale, alors une discontinuité est détectée et les coefficients sont réajustés progressivement. Toutes ces techniques de débruitage concernent souvent le post-traitement d'images fixes mais peuvent être étendues au domaine spatio-temporel.

2.1.3 Transcodage

Les transcodeurs semblent très prometteurs dans le contexte de globalisation de réseaux hétérogènes car les terminaux ont chacun leur format vidéo. Il convient alors d'effectuer des conversions au niveau des serveurs sans exiger d'eux de lourds traitements. La taille d'une image est un paramètre que l'on désire souvent modifier. On procède traditionnellement ainsi par un sur-échantillonnage pour agrandir d'un facteur *M*, puis par un sous-échantillonnage pour diminuer d'un facteur *N* et redimensionner l'image d'un facteur *M/N*. À noter que ce processus nécessite un filtrage d'interpolation et d'anti-repliement (Park et al., 2003). Le rétrécissement ne pose pas de problème, d'où les essais d'implémentation complète pour le standard *MPEG* (Merhav et al, 1997), (Fung et al., 2006). Pour l'agrandissement, ajouter des coefficients à zéros pour les hautes fréquences n'est pas satisfaisant et peut entraîner des discontinuités entre les blocs, ce qui peut être évité en utilisant la corrélation entre des blocs voisins (Goto et al., 2008). Améliorer la résolution demeure encore un grand champ d'investigation même dans le domaine spatial. Il est possible d'exploiter l'interpolation temporelle à partir de multiples images de référence pour créer de bons détails (Martins et al., 2002).

Il est parfois nécessaire de réduire encore plus le débit pour s'adapter aux restrictions d'un canal particulier. Ceci peut être obtenu en modifiant le format de la chrominance, notion abordée dans le chapitre 3. Toujours dans le domaine *DCT*, il est possible de renforcer la quantification comme (Assunçao et al., 1997) ou bien de supprimer tout simplement certaines images, ce qui nécessite de recoder les mouvements comme (Fung et al., 2004).

D'une manière générale, on peut imaginer toutes les transformations géométriques (linéaires) possibles appliquées aux coordonnées des pixels. (Ishwar et al., 1998) montre comment réaliser simplement les retournements horizontal et vertical, les rotations aux multiples de 90 degrés près et les inclinaisons (deux inclinaisons peuvent donner une rotation d'angle arbitraire). Bon nombre de ces opérations nécessitent l'inversion de coefficients *DCT*. Il faut bien entendu les vecteurs de déplacement pour la vidéo mais l'article ne semble pas réellement s'intéresser à ce format.

2.2 Composition des deux médias dans le Domaine *DCT*

Certaines des informations qui suivent portent sur l'algorithme de codage de *MPEG-2* ou sur la solution technique du projet. Ces éléments sont brièvement décrits mais les chapitres concernés les reprennent plus en détails.

2.2.1 Problème d'alignement de structures en blocs

Superposer des objets (texte, image, vidéo) pour qu'ils s'intègrent dans l'image impose souvent une précision au pixel près, ce qui est très simple dans le domaine spatial mais pas dans le domaine de transformée. Donc la première contrainte vient de la structure en blocs *DCT* imposée par la norme de codage. Il y a de fortes chances pour qu'un objet codé *DCT* à superposer ait une position ou des dimensions qui ne soient pas multiple de 8, ce qui induit un décalage entre la structure de blocs de l'arrière plan et celle de l'objet à ajouter. Chang et Messerschmitt (Chang et al,. 1995) sont les premiers à proposer une méthode pour réarranger la structure de l'image désignée comme premier plan pour qu'elle concorde avec celle de l'arrière plan. En effet, il est obligatoire et logique de conserver la structure du média «contenant» plutôt que celui «contenu». Un bloc *DCT* du premier plan en chevauche généralement quatre de l'arrière plan. L'algorithme de ces auteurs permet d'extraire quatre contributions du bloc superposé et de les intégrer dans les quatre blocs correspondants de l'arrière plan. Les produits matriciels nécessaires pour cette opération sont transposables dans le domaine fréquentiel grâce à l'orthogonalité de la *DCT*.

2.2.2 Problème de dépendance temporelle entre les images

Le principe de codage temporel d'une séquence codée *MPEG* est expliquée en détail dans le chapitre 3. Une image est codée à partir des éléments obtenus par une estimation de mouvements (EM). Il s'agit d'analyser les déplacements des objets de l'image courante par rapport à une (ou plusieurs) image(s) de référence. À l'inverse, le processus de décodage effectue une compensation de mouvement (CM) des objets de l'image de référence pour reconstruire l'image courante. Cette dépendance temporelle pose problème car modifier une image de référence peut propager une erreur dans les images qui en dépendent.

L'idée primordiale est toujours de travailler dans le domaine *DCT* en profitant des propriétés de la transformée. Par contre, la CM et l'EM opèrent dans le domaine spatial et ne présentent pas ces propriétés d'orthogonalité et de linéarité. (Chang et al,. 1995) proposent de générer le domaine *MC-DCT* (*Motion Compensated DCT*). Pour ce faire, ils modifient le processus de décodage partiel en effectuant la CM sur des images codées *DCT*. Le résultat est une séquence d'images reconstruite mais avec une structure de blocs de coefficients *DCT* librement modifiables. Remarque importante : la CM utilise des vecteurs de mouvement pour aller chercher un bloc de prédiction dans l'image de référence. Or cette référence a maintenant aussi une structure en blocs de 8x8 pixels et les composantes de ce déplacement sont rarement des multiples de 8. Donc le problème de réalignement de structures se pose à nouveau. Cette fois-ci, il faut extraire les contributions des 4 blocs de référence pour former le bloc de prédiction.

2.2.3 Problème de découvrement

Lors de la superposition d'une image dans la zone 1 sur la Figure 2-1, les objets en arrière plan sont écrasés, ce qui ne provoque aucune erreur dans l'immédiat. Or ces objets sont souvent mobiles et ils doivent parfois réapparaître sur un côté de l'image superposée (zone 2). Le décodeur ne sera pas en mesure de retrouver les objets qui ont été supprimés au moment de la superposition dans l'image de référence. C'est pour cette raison que la zone 2 de la Figure 2-1 située autour de la zone de superposition est directement affectée par ce problème de découvrement de l'arrière plan. On devrait y voir des morceaux de l'image du premier plan se déplacer selon les mouvements de l'arrière plan. De plus, cette erreur peut encore se propager plus loin, dans la zone 3 pour des images suivantes, si les objets originaux de l'arrière plan continuent leur déplacement. (Chang et al,. 1995) souhaitent recréer l'arrière plan pour tous les blocs concernés de la zone 2 en ré-estimant le mouvement lors de l'étape de recodage. L'EM est l'opération la plus coûteuse de l'ensemble codeur/décodeur. La recherche se limite alors à seulement deux positions : la première en conservant la composante verticale du vecteur de mouvement original, et la deuxième en conservant la composante horizontale. L'innovation repose sur le fait de réaliser l'EM dans le domaine de transformée.

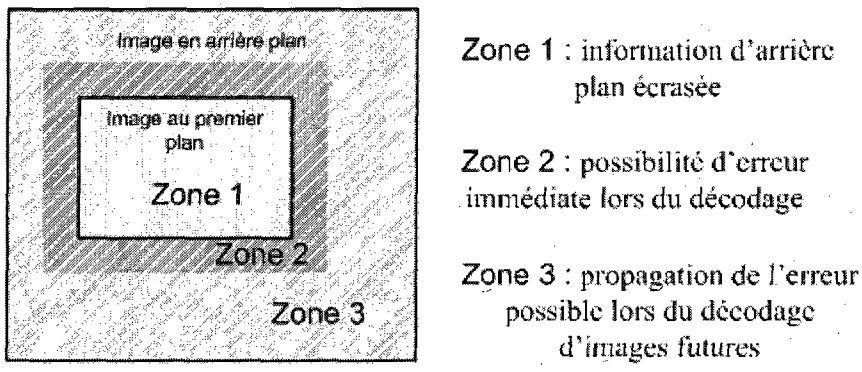


Figure 2-1 Traitement par zone du problème de découvrement de l'arrière plan

2.2.4 Cas particulier de l'insertion d'image fixe

La superposition semi-transparente d'un bloc de pixels est obtenue par la combinaison linéaire de ce bloc et de celui superposé. La propriété linéaire de la *DCT* est exploitable pour transposer la combinaison linéaire dans le domaine de transformée. Il est également possible d'ajuster la transparence pour chaque pixel de l'image à insérer en le pondérant par une valeur de masque. Ceci permet d'insérer une image de forme arbitraire telle qu'un sous-titre. (Jòsson et al., 1997) réussissent à insérer ce genre d'image dans le domaine *DCT* grâce à la théorie de multiplication/convolution.

Par définition, une image n'a aucun vecteur de mouvement. Lors de l'incrustation, une méthode intuitive consiste à annuler les vecteurs de mouvement dans la zone de l'image. Cependant, cela ne permet pas de s'affranchir des problèmes de découvrement. (Panusopose et al., 2001) préfèrent conserver le vecteur de mouvement original. Leurs travaux débouchent sur une architecture de transcodeur en cascade avec un décodeur suivit d'un module d'insertion puis d'un codeur qui réutilise ce vecteur. Le problème est que le flux binaire est complètement décodé. (Roma et al., 2002) reprennent à la fois les concepts de masquage pour insérer un logo de forme arbitraire et l'architecture de transcodeur mais en l'adaptant pour opérer dans le domaine *MC-DCT*.

2.3 Commentaire et proposition

Les travaux de (Chang et al,. 1995) et de (Roma et al., 2002) s'imposent comme les travaux les plus aboutis en matière de composition de plusieurs médias images. Il s'agit à présent de distinguer les éléments exploitables de ceux inutiles pour élaborer la stratégie du projet.

2.3.1 La précision au pixel près

La méthode de (Chang et al,. 1995) est ingénieuse, car elle exploite bien les propriétés mathématiques de la *DCT* pour effectuer des traitements au pixel près malgré la structures en blocs *DCT*. Cet outil d'extraction/déplacement de portion de bloc constitue un élément clé de ce projet à la fois pour positionner le logo mais aussi pour les manipulations dans le domaine *MC-DCT*. L'intégration d'objets de forme arbitraire semble être réalisée avec succès par (Roma et al., 2002) mais ce critère de forme ne fait pas partie des spécifications du projet.

2.3.2 Domaine *MC-DCT*

(Chang et al,. 1995) sont aussi les premiers à avoir surmonté les difficultés liées au codage temporel et à transposer la CM et l'EM dans le domaine *DCT*. Le processus semble exhaustif car il induit beaucoup de produits matriciels, mais (Merhav et al., 1996) assurent qu'il n'est pas très coûteux en termes de calcul car la plupart des coefficients *DCT* sont égaux à 0, et donc, beaucoup de multiplication sont annulées. C'est pour cette raison que les transcodeurs de (Merhav et al., 1997), (Fung et al., 2004), (Fung et al., 2006) et (Roma et al., 2002) ont repris la technique.

2.3.3 Proposition

Cependant, un point de la méthode de (Chang et al,. 1995) est tout à fait critiquable. Les données (vecteur de mouvement et erreur de prédiction) du premier plan écrasent celles de l'arrière plan pour (Chang et al,. 1995). Le traitement dans la zone de découvrement de l'arrière plan nécessite de le recréer complètement par une nouvelle EM, très coûteuse en calcul, surtout dans le domaine *MC-DCT*. (Roma et al., 2002) et (Panusopose et al., 2001) ont bien saisi l'intérêt de conserver le vecteur de mouvement. Cependant, leur structure de

transcodeur en cascade insère le logo au niveau de l'image reconstruite *MC-DCT* et le recodage nécessite une seconde CM, toujours dans le domaine *DCT*. De la même manière, l'algorithme mis en œuvre dans ce projet conserve les vecteurs de mouvement car ceci permet d'économiser une EM lors du recodage. Ce choix est aussi justifié par la volonté de conserver les mouvements de l'arrière plan dans le cas d'une superposition semi-transparente du logo. Ensuite, plutôt que d'appliquer les transformations sur l'image reconstruite *MC-DCT*, il est préférable de se concentrer seulement sur l'erreur de prédiction qui se présente dans le flux binaire sous forme *DCT* et d'y apporter les corrections nécessaires. Pour conclure, là où (Chang et al,. 1995) mettent en œuvre une CM et une EM pour insérer une image et (Roma et al., 2002) mettent deux CM, la méthode décrite dans le chapitre 4 n'a a priori besoin de rien. Cependant, une CM dans le domaine *DCT* est parfois nécessaire pour bien corriger l'erreur de prédiction.

3 LE STANDARD MPEG-2

3.1 Présentation

3.1.1 Choix du standard

Dans le cadre de ce projet, le standard *MPEG-2* est utilisé comme norme de compression pour le signal vidéo. Bien qu'il soit relativement ancien (1994), ce standard est encore couramment utilisé pour le stockage de données vidéo sur support *DVD* et pour la transmission de la télévision numérique par satellite, câble, Internet. Il permet d'obtenir un débit de l'ordre de 10 Mbit/s pour les dimensions standard (*SD*), ce qui le rend moins performant que son successeur (*MPEG-4*), mais il reste plus simple à comprendre. La norme *MPEG-2* prend en compte les spécifications du système (*ISO/CEI 13818-1*) pour la synchronisation, la transmission et le stockage, et les spécifications pour le codage (*ISO/CEI 13818-2 et 3*) (ISO/IEC, 1994). Cette dernière partie regroupe le codage des données vidéo et celui des données audio. Pour ce projet, il semble justifié de se concentrer uniquement sur le codage de la vidéo et d'oublier le reste. Les spécifications (ISO/IEC, 1994) se réfèrent à la méthode de compression. Sans trop rentrer dans les détails, il convient de donner quelques éléments qui sont utiles à la compréhension du fonctionnement du codec et du travail effectué. C'est aussi l'occasion d'exposer la configuration dans laquelle est utilisé le codec.

3.1.2 Profils et niveaux

Pour couvrir plusieurs applications avec chacunes leurs contraintes, la norme *MPEG-2* prévoit plusieurs profils d'utilisation avec pour chacun un niveau paramétrable. Comme indiqué dans le tableau 3-1, les profils et leurs niveaux concernent surtout les dimensions des images, le format de la chrominance et donc le débit du flux de données. On trouve également la notion de profils échelonnables. Il s'agit de fonctions supplémentaires qui génèrent, en plus du flux binaire de base, d'autres informations complémentaires qui

permettent lors du décodage d'améliorer la résolution spatiale (profil ajustable spatialement) ou simplement d'améliorer la qualité de l'image (profil ajustable en rapport signal sur bruit). Cette notion est écartée dans le cadre de ce projet car on se restreint au profil haut avec le niveau principal qui permet de travailler avec un format de chrominance 4 :2 :2 et avec un débit raisonnable pour la télédiffusion.

TABLEAU 3-1CONFIGURATIONS DU CODEC MPEG-2

Profil		Simple	Principal	Ajustable en RSB	Ajustable spatialement	haut
Niveau	Haut		4:2:0 1920x1152 80 Mbps			4:2:0/4:2:2 1920x1152 100 Mbps
	Haut - 1440		4:2 0 1440x1152 60 Mbps		4:2:0 1440x1152 60 Mbps	4:2:0/4:2:2 1440x1152 80 Mbps
	Principal	4:2:0 720x576 15 Mbps	4:2:0 720x576 15 Mbps	4:2:0 720x576 15 Mbps		4:2:0/4:2:2 720x576 20 Mbps
	Bas		4:2:0 352x288 4 Mbps	4:2:0 352x288 4 Mbps		

3.2 Format des données traitées

3.2.1 Espace de couleur

Tout d'abord, il est important de connaître le format des données, en commençant par l'espace de couleurs. L'espace RVB est le plus intuitif car la synthèse d'une couleur est obtenue par une combinaison linéaire des trois couleurs primaires Rouge, Vert et Bleu. Depuis les premières transmissions de signaux vidéo analogiques, on a plutôt cherché à séparer l'information de luminance, notée Y, et celles de chrominance, notées U et V.

L'espace YUV est encore utilisé pour la transmission de la vidéo numérique, donc dès l'entrée du codeur *MPEG* jusqu'à la sortie du décodeur. Par contre, l'espace RVB est nécessaire pour visualiser ces images en amont et en aval de la chaîne de transmission. L'équation (3.1) permet la conversion d'un pixel YUV en un pixel RVB.

$$\begin{bmatrix} R \\ V \\ B \end{bmatrix} = \begin{bmatrix} 1 & -0.00004 & 1.14 \\ 1 & -0.395 & -0.581 \\ 1 & 2.032 & -0.0005 \end{bmatrix} \times \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \tag{3.1}$$

3.2.2 Format de la chrominance

Quand une image se présente sous la forme des composantes YUV, il est possible d'exploiter les propriétés du système visuel humain pour éliminer une partie de l'information. En effet, pour les images naturelles les deux composantes U et V sont en général, à bande limitée. Ce qui veut dire que dans le contexte de compression, il est possible de perdre un peu de résolution spatiale sur les deux composantes de chrominances U et V en les filtrant et les sous-échantillonnant sans pour autant que l'œil humain sans aperçoive. Ceci donne lieu à un choix de trois formats de chrominance (4:4:4, 4:2:2, 4:2:0) proposé par la norme *MPEG-2*, dépendamment de l'application. Le premier format garde la pleine résolution mais reste très peu utilisé. Pour une application de télédiffusion, c'est le format 4:2:2 qui est employé. Il s'agit alors de supprimer une colonne de pixels sur deux. Enfin, le format 4:2:0 consiste à supprimer à la fois une colonne et une ligne sur deux. Il trouve son application dans la téléconférence et supporte assez mal les encodages et décodages répétés. La Figure 3-1 illustre un exemple de changement d'espace de couleur et une mise au format 4:2:2.

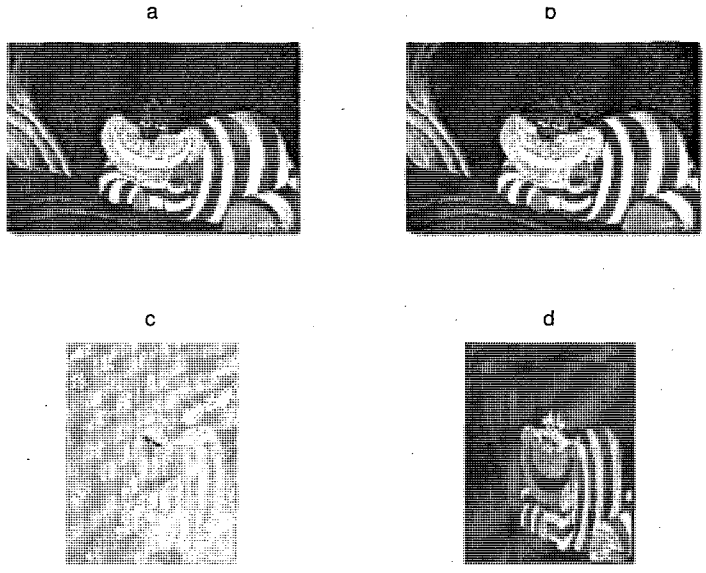


Figure 3-1 Décomposition d'une image RGB (a) dans le nouvel espace de couleur Y (b), U(c), V (d) et mis au format 4:2:2

3.2.3 Entrelacé / progressif

La notion d'entrelacement datant de la télédiffusion analogique est toujours en vigueur pour la vidéo numérique. On rappelle que l'entrelacement consiste à transmettre successivement deux trames par image (la première contient les lignes paires et l'autre, les lignes impaires) au lieu de transmettre une image en entier d'un seul coup dans le cas d'une image progressive. Cette technique exploite la persistance rétinienne pour doubler la fréquence de balayage sans augmenter le volume d'information. L'effet de papillotement engendré par une séquence d'images progressives est alors évité. Cependant, la structure des images est progressive pour ce projet pour alléger le code, notamment dans l'adressage de ses structures de données. De plus, l'entrelacement n'apportera pas d'éléments nouveaux à l'étude.

3.2.4 Structure macrobloc

Les images ne sont alors plus stockées ligne par ligne, comme pour un fichier *bitmap* ; elles sont représentées dans le flux de données sous forme de blocs de 8x8 pixels, comme pour un fichier codé avec la norme JPEG. MPEG-2 va même plus loin en rassemblant quatre blocs Y et plusieurs blocs U et V dans une structure appelée macrobloc. Chaque macrobloc comprend

donc 16x16 pixels Y et le nombre de pixels de la chrominance dépend du choix de sous-échantillonnage de ses 2 composantes. Dans le cas du 4 :2 :2, les pixels des composantes U et V sont sous-échantillonnés horizontalement par un facteur de 2. Ainsi, chaque macrobloc comprend quatre blocs Y, deux blocs U et deux blocs V qui sont numérotés comme sur la Figure 3-2, il s'agit de l'ordre d'apparition des blocs dans le fichier. Enfin, plusieurs macroblocs successifs dans le sens horizontal sont regroupés pour former une tranche. L'intérêt de ce regroupement est expliqué dans la section 3.4.

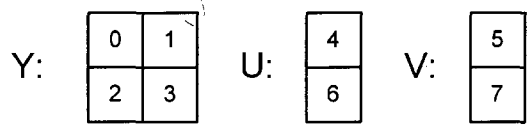


Figure 3-2 Structure macrobloc YUV

3.3 Principe de codage

3.3.1 Prédiction temporelle

La plupart du temps, des images successives d'une séquence sont similaires et c'est pour cette raison que l'on cherche à éviter cette redondance temporelle en codant une image courante par prédiction temporelle. Les premières images, dites *intra* (*I*), sont codées comme des images fixes tel qu'avec la norme *JPEG*. Elles servent de références pour la prédiction par estimation de mouvement des images, dites *inter* (*P* pour *prediction, B* pour *bidirectionnal*). La Figure 3-3 montre le sens de codage/décodage d'une séquence qui est le même que l'ordre d'apparition de ces images dans le flux de données *MPEG-2*. Les images *P* sont codées et interpolées au niveau du décodeur par prédiction avant, c'est-à-dire à partir d'une image *I* ou d'une autre image *P* située dans la passé dans la séquence originale. Les images *B* sont interpolées par prédiction bidirectionnelle, c'est-à-dire à partir de deux images de référence (*I* ou *P*) une dans le passé et l'autre dans le futur. L'insertion régulière d'image intra permet une segmentation temporelle précise d'une séquence en *GOP* (*Group Of Pictures*). Il existe beaucoup de choix possibles pour la structure de *GOP* ; une qui est couramment employée est la suivante *IBBPBB IBBPBB*...

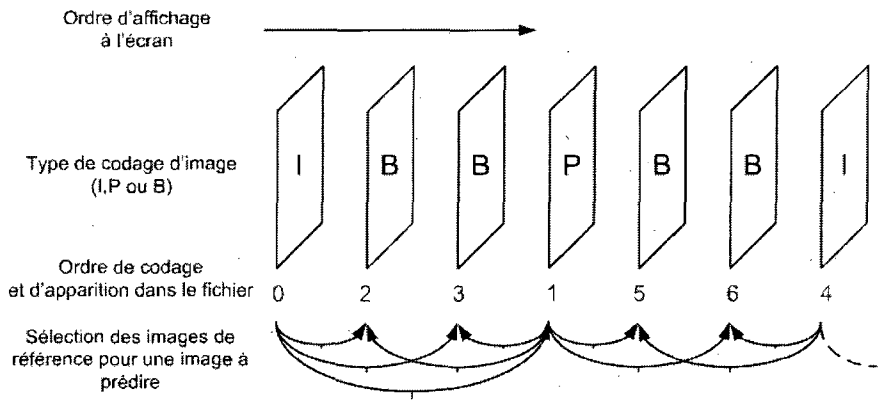


Figure 3-3 Sens de codage/décodage d'une séquence *I B B P B B I*...

3.3.2 Algorithme de codage

L'algorithme illustré par la figure 3-4 recherche, pour un macrobloc en entrée, le macrobloc qui lui ressemble le plus dans une (ou des) image(s) de référence. Ce processus appelé l'estimation de mouvement fournit un vecteur de mouvement et le macrobloc de référence s'appelle la prédiction. Les images de références (*I* ou *P*) ont été codées auparavant puis décodées et enfin stockées dans une mémoire tampon, visible au centre de la figure 3-4.

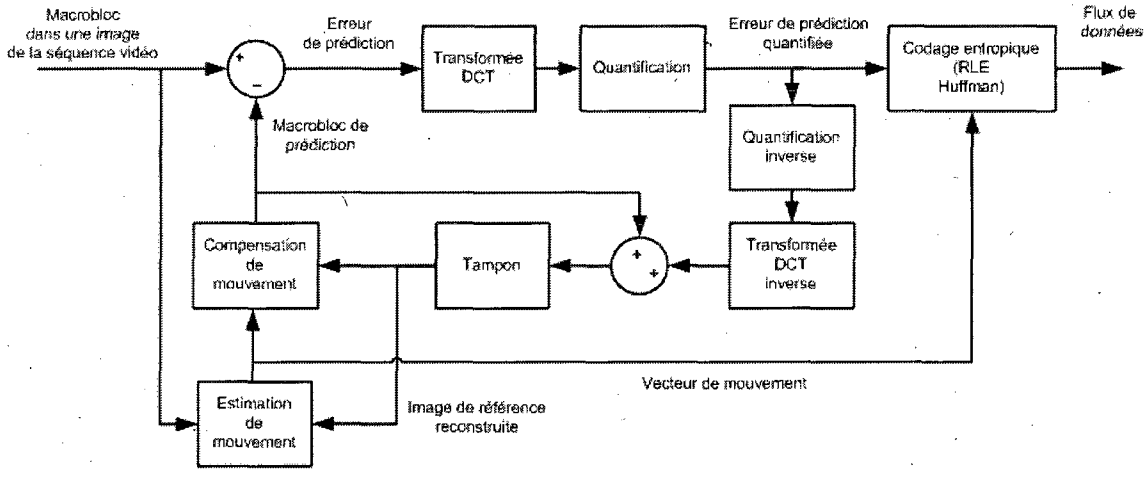


Figure 3-4 Schéma fonctionnel du codeur *MPEG-2*

Il y a de fortes chances pour que le macrobloc de prédiction et le macrobloc courant soient légèrement différents, donc il faut également transmettre leur différence, notée erreur de prédiction. Les blocs d'erreur, contenant de plus petites valeurs que les blocs originaux, passent par une transformation *DCT-II* (*Discrete Cosine Transform Type 2*). L'équation (3.2) montre comment obtenir le bloc B_{DCT} dans le domaine fréquentiel (u,v) à partir du bloc B défini dans le domaine spatial (x,y).

$$B_{DCT}(u,v) = \frac{2}{N} \cdot C(u) \cdot C(v) \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} B(x,y) \cdot \cos\left(\frac{(2x+1) \cdot u\pi}{2N}\right) \cdot \cos\left(\frac{(2y+1) \cdot v\pi}{2N}\right)$$
$$\text{avec } N = 8, \quad C(u), C(v) = \begin{cases} 1/\sqrt{2} & \text{pour } u, v = 0 \\ 1 \text{ sinon} \end{cases} \tag{3.2}$$

Ensuite tous les coefficients *DCT* passent par une quantification scalaire dont le pas varie en fonction de la fréquence et selon le type de bloc (intra/inter, luminance/chrominance). À titre d'exemple, l'équation (3.3) regroupe ces pas de quantification dans une matrice, pour la composante Y des images *intra* (ISO/IEC, 1994). Les pas en haut à gauche, plus fins (M_{QIntra} (0,0) = 8), conservent mieux les basses fréquences (BF) par rapport aux pas en bas à droite correspondant aux plus grandes fréquences (HF). En effet, la majeure partie de l'énergie d'une image naturelle est contenue dans les coefficients BF et, à l'inverse, quantifier grossièrement les détails correspondants aux HF est moins dérangeant pour le système visuel.

$$M_{QIntra} = \begin{bmatrix} 8 & 16 & 19 & 22 & 26 & 27 & 29 & 34 \\ 16 & 16 & 22 & 24 & 27 & 29 & 34 & 37 \\ 19 & 22 & 26 & 27 & 29 & 34 & 34 & 38 \\ 22 & 22 & 26 & 27 & 29 & 34 & 37 & 40 \\ 22 & 26 & 27 & 29 & 32 & 35 & 40 & 48 \\ 26 & 27 & 29 & 32 & 35 & 40 & 48 & 58 \\ 26 & 27 & 29 & 34 & 38 & 46 & 56 & 69 \\ 27 & 29 & 35 & 38 & 46 & 56 & 69 & 83 \end{bmatrix} \tag{3.3}$$

D'un côté, les coefficients *DCT* quantifiés sont rangés de manière à regrouper les valeurs qui ont plus de chance d'être nulles, les coefficients HF. Le codage entropique de *Huffman* (appelé aussi *VLC* pour *Variable Length Code*) est visible sur la figure 3-4 comme étant la dernière étape du codage. Il exploite la redondance des valeurs nulles pour former le flux de bits plus compact et sans perte. Au passage, le but général du *VLC* est d'attribuer un mot court à une valeur qui a une forte probabilité d'apparition, et inversement, un mot long une valeur rare. Des informations supplémentaires sur le contenu des blocs seront apportées à la section 3.4.6. De l'autre côté, ces mêmes coefficients *DCT* quantifiés subissent les opérations inverses pour former une nouvelle image de référence dans la mémoire tampon, si nécessaire (les *B* ne servent jamais de référence).

3.3.3 Du côté du décodeur

La Figure 3-5 montre le décodeur qui réalise les opérations inverses à partir du flux de données. Il s'agit de retrouver le vecteur de mouvement afin d'aller chercher le macrobloc de prédiction dans l'image de référence par compensation de mouvement. Ensuite, les blocs d'erreur de prédiction sont décodés (décodage entropique, quantification inverse, *DCT* inverse) avant d'être ajoutés aux blocs de prédiction. Il en résulte une nouvelle image reconstruite qui peut servir de référence pour la suite, ici aussi stockée dans une mémoire tampon.

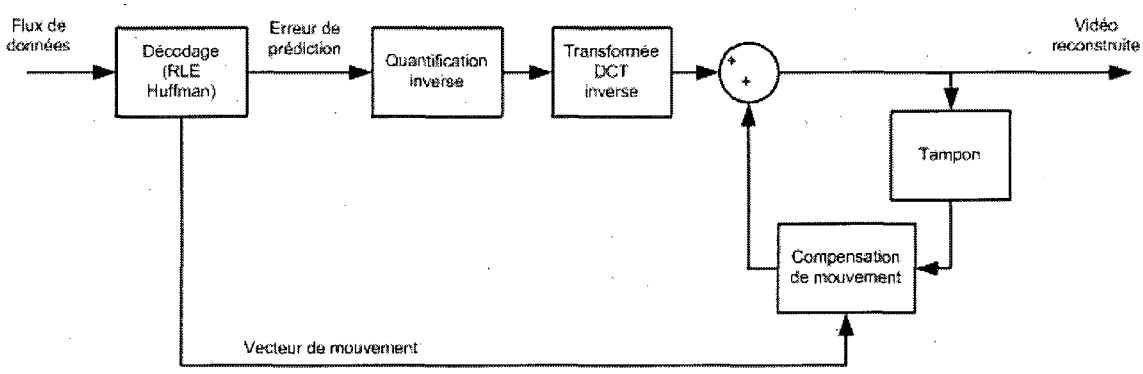


Figure 3-5 Schéma fonctionnel du décodeur *MPEG-2*

3.4 Description du flux binaire

La Figure 3-6 décrit la syntaxe d'un fichier imposée par la norme *MPEG-2*. Les principaux paramètres contenus dans les entêtes sont ensuite détaillés s'ils sont nécessaires pour l'application. Certains paramètres en gras et italique sont utilisés par l'application. Parmi eux, certains servent à se localiser dans le temps et dans l'image, d'autres à déterminer les paramètres de quantification et des informations essentielles au bon fonctionnement de l'application.

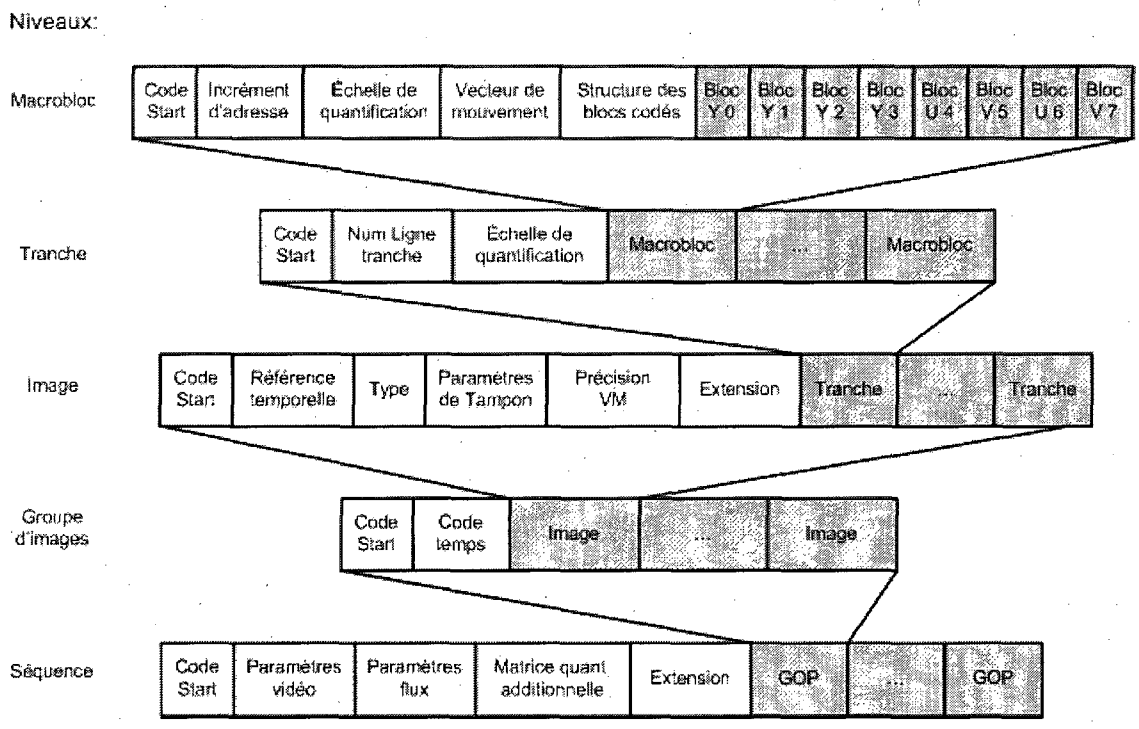


Figure 3-6 Syntaxe d'un flux de données codé MPEG-2

3.4.1 Entête d'une séquence :

- ***Paramètres vidéo*** : largeur, hauteur, nombre d'images par seconde, rapport des dimensions de l'image affichée, rapport de dimensions des pixels affichés. Seules la largeur et la hauteur de l'image sont utiles pour se repérer dans l'image (Exemple de format standard : ITU601 (720 x 480)).

- *Paramètres du flux de données* : débit, taille du tampon. Ceci peut être utile si le projet est amené à être implémenté matériellement.
- *Matrice de quantification additionnelle* : dans le cas où des matrices de quantification personnalisées sont utilisées. Elles sont inutiles puisque c'est la matrice de quantification par défaut qui est employée.
- ***Extension*** : paramètres supplémentaires : format de la chrominance, mode progressif ou entrelacé, profile et niveau du codeur selon l'application envisagée.

3.4.2 Entête d'un *GOP*

- *Code Temporel* : heure, minutes et secondes du début du groupe. Cette information est intéressante dans le cas de l'incrustation de sous-titre, mais elle n'est pas utile ici dans la mesure où les paramètres de composition sont constants au cours du temps.

3.4.3 Entête d'une image:

- *Référence temporelle* : il y a une différence entre l'ordre de codage des images et l'ordre d'affichage du fait de la prédiction bidirectionnelle. Là encore, la dimension temporelle n'est pas très utile puisque le fichier utilisé ne contient pas d'images *B*.
- ***Type*** : intra, prédiction ou bidirectionnel. Ceci est indispensable car les données à transmettre sont fondamentalement différents entre une image *intra* et une *inter*.
- *Paramètre de tampon* : indique l'état supposé du tampon juste avant le décodage de l'image.
- *Précision VM:* indique si la précision des vecteurs de mouvement est au demi-pixel près. Cette information est précieuse car une grosse partie du projet gère les problèmes liés à la prédiction.
- *Extension* : autres paramètres d'encodage.

3.4.4 Entête d'une tranche

- ***Numéro Ligne de la tranche*** : Une tranche est un assemblage horizontal de macroblocs et chaque nouvelle ligne de macroblocs donne une nouvelle tranche. Donc il suffit d'avoir le numéro de la ligne et les dimensions de l'image pour avoir l'adresse du premier macrobloc de la tranche.
- ***Facteur de quantification*** : facteur influençant la finesse de la quantification des coefficients *DCT*, selon le débit imposé. Ce paramètre ne doit pas être modifié mais il sert pour le recodage d'un bloc.

3.4.5 Entête d'un macrobloc

- ***Incrément d'adresse*** : L'adresse des macroblocs initialisée à 0 en haut à gauche de l'image s'incrémente à chaque déplacement vers la droite et à chaque changement de ligne (tranche). Ce n'est pas l'adresse qui est codée mais l'incrément d'adresse. Sa valeur est normalement 1 sauf si des macroblocs n'ont pas été codés, ce qui arrive quand des zones de l'image restent fixes au cours du temps. L'adresse et donc l'incrément sont précieux pour se situer dans l'image.
- ***Modes macrobloc*** : apporte des informations sur la prédiction et la compensation de mouvement (avant, arrière, basée sur un format entrelacé/progressif).
- ***Facteur de quantification*** : met à jour le facteur de quantification si besoin. On rappelle que cette information nous est utile pour le codage de nos blocs.
- ***Vecteurs de mouvement*** : indique si le macrobloc utilise une compensation de mouvement et apporte des informations sur le vecteur qu'il faut exploiter et/ou modifier car le logo est immobile et indépendant du mouvement dans la séquence.
- ***Structure des blocs codés*** : distingue les blocs du macrobloc qui ont été codés de ceux qui ne l'ont pas été. Tous les bits de ce paramètre sont à 1

(tous les blocs sont codés) pour une image *intra*, mais il faut y faire plus attention pour les images *inter*.

3.4.6 Contenu d'un bloc

Les informations qui suivent dans le flux sont les données propre à chacun des blocs qui sont ordonnés comme sur la Figure 3-2. Il est très important de connaître le contenu d'un bloc pour pouvoir le décoder (même partiellement pour ce projet) et surtout pour pouvoir insérer correctement un bloc logo. Ces blocs contiennent les 64 coefficients *DCT* quantifiés des blocs 8x8 pixels.

La première remarque concerne les coefficients *DC* (*Direct Current* pour la fréquence spatiale nulle) de la luminance et de la chrominance pour les images *intra*. La précision de ces coefficients a un impact fort sur la qualité de l'image reconstruite. La première information du flux d'un bloc est le nombre de bits sur lequel la *DC* est codée. Ensuite, une modulation par impulsion et codage différentiel (ou *DPCM* pour *Differential Pulse Code Modulation*) transmet la différence entre la *DC* du bloc courant et celle du précédent. Deux blocs successifs sont supposés avoir des *DC* similaires, surtout dans les zones unies de l'image. Ainsi, ce codage exploite la redondance spatiale en fournissant une valeur qui prend moins de place qu'un coefficient *DCT* en tant que tel. Ce processus est appliqué à chacune des composantes de couleur. La Figure 3-7 complète l'explication en donnant l'ordre des blocs pris en charge par le codage *DPCM* pour le cas de la luminance. On retrouve l'ordre d'apparition des blocs Y dans le flux de données tel que l'indique la figure 3-2.

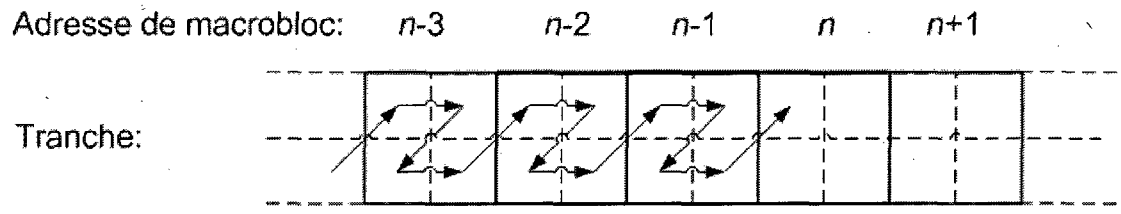


Figure 3-7 Parcours du codage *DPCM* des coefficients *DC* pour les blocs Y jusqu'au bloc 0 du macrobloc *n*.

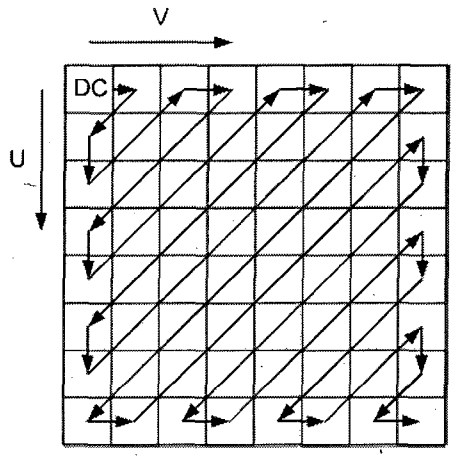


Figure 3-8 Sens de parcours des coefficients *AC* en « zigzag » pour un bloc, depuis la *DC* jusqu'au coefficient *AC* de la plus haute fréquence spatiale.

En ce qui concerne maintenant tous les blocs *DCT* quantifiés, il faut à présent coder les 63 coefficients *AC* (*Alternative Current*) correspondants de fréquences non nulles. Dans le cas d'une image naturelle, plus la fréquence spatiale augmente, plus les coefficients *AC* quantifiés ont de chance d'être nuls. Or, d'un point de vue compression, une grande série de zéros successifs apporte un avantage que le codage entropique de *Huffman* (à code de longueur variable) exploite. C'est pour cette raison que les coefficients *AC* sont parcourus soit de façon dite « alternée » soit en « zigzag » comme indiqué sur la Figure 3-8. La deuxième option est sélectionnée car elle est plus répandue.

3.5 Conclusion

Pour résoudre la problématique, il y a donc plusieurs facteurs qui entrent en jeu. Certains problèmes peuvent déjà être identifiés : la structure en blocs *DCT* va complexifier le traitement au pixel près, et surtout la prédiction temporelle devrait être le principal souci. En effet, si modifier la première image du *GOP* (*intra*) ne semble pas causer de difficulté, modifier les images *inter* est une tout autre affaire car elles dépendent toutes de la première. C'est-à-dire qu'il faut trouver un moyen de gérer les mouvements codés en jouant sur le

vecteur de chaque macrobloc concerné ou leurs erreurs de prédiction. D'autres détails techniques ajoutent quelques subtilités à l'algorithme. Il s'agit de la prise en charge :

- des trois formats de chrominance;
- du codage *DPCM* pour les coefficients *DC* des blocs *intra* et les vecteurs de mouvement;
- de la précision au demi-pixel de ces mêmes vecteurs;
- des blocs non codés, etc.

En revanche, d'autres considérations techniques sont mises de côté pour alléger le projet. À titre d'exemple, le mode entrelacé, bien que souvent utilisé, n'apporte à l'étude aucun élément de compréhension et alourdirait le code. Ensuite, tous les modes opératoires (profils d'utilisation et niveaux) ne peuvent être abordés puisque certains complexifient considérablement le décodage en ajoutant des post-traitements. C'est pour cette raison que les modes échelonnables sont écartés. Le chapitre 4, c'est-à-dire l'article qui condense toute la solution technique et dont l'évaluation est en cours, détaille un peu plus les conditions expérimentales.

4 ARTICLE SUR LA COMPOSITION D'UN LOGO DANS LE DOMAINE *DCT*

4.1 Avant-propos

Auteurs : Patrick Keroulas et Chon Tam LeDinh

Référence de publication : *IEEE Transactions on Broadcasting*

Date de soumission : Février 2009.

État de publication : en cours d'évaluation.

Titre en français : Composition d'un logo dans le domaine *DCT* pour le standard *MPEG-2*.

Résumé en français : Cette publication présente une technique de traitement vidéo directement applicable sur un flux binaire codé, permettant de passer outre les processus de décompression et recompression complets. Il s'agit de superposer un logo dans une séquence d'images codée *MPEG-2*. La Transformée en Cosinus Discrets (*DCT*) utilisée par ce standard présente quelques propriétés mathématiques intéressantes qui permettent de gérer l'opacité du logo et sa position au pixel près. Cet algorithme traite du problème de recouvrement/découvrement de l'arrière plan grâce à une compensation de mouvement locale, dans le domaine *DCT* (*IMC-DCT*).

Termes clés : composition, domaine *DCT*, compensation de mouvement, MPEG-2, compression vidéo

Logo compositing in the DCT domain for MPEG-2 Standard

Patrick Keroulas and Chon-Tam Le Dinh, *Member*, IEEE

Abstract— This paper presents a video processing technique which is applied on a coded bit stream in order to avoid the complete uncompression/recompression processes. It consists of overlapping a logo on a MPEG-2 coded moving picture. The Discrete Cosine Transform (DCT) used by this standard presents interesting mathematical properties that provide a pixel-accurate position for the logo and an adjustable opacity. The algorithm features the covering/uncovering of the original background thanks to a local inverse motion compensation in the DCT domain (IMC-DCT).

Index Terms — compositing, DCT domain, motion compensation, MPEG-2 Standard

4.2 Introduction

With the emergence of digital video on broadband networks, coding standards have been developed to decrease data rate. Processing is needed during the transmission on the network or the storage on a database, meaning after the coding process. These operations are usually applied in the spatial domain and are computationally demanding. Recent works investigated on the transposition of these operations to the compressed domain. Its main purpose is the reduction of the computational complexity because it needs no decompression or recompression and thus the number of coefficients to be modified is smaller.

As an example of basic processing, a filter operation can be applied in the DCT domain by transposing the convolution in the spatial domain to a multiplication in the frequency domain, as in (Merhav et al., 1999) and (Changhoon, 2004). Many geometric transformations can also be easily transposed such as image flipping, rotation and scaling (Chen et al., 1998) and

(Martucci, 1995). These operations can be implemented in concrete applications like the transcoding (Ghanbari et al., 1997) and noise or blocking artifact reduction (Triantafyllidis et al., 2002).

In this work, we present a method to overlap a logo in a MPEG-2 coded sequence. (Roma et al., 2002) showed how to insert an irregular-shaped logo thanks to a mask and the Multiplication-Convolution Theorem, but we prefer to focus on the rectangular object compositing which was largely discussed by (Chang et al., 1995). These authors supply the tools that help to manage the two compositing options: the semi-transparent mode and the pixel accurate position of the logo. The first section explains how these two features derive from the linear and orthogonal properties of the DCT. With the dependence between the frames in the MPEG-2 coding algorithm (ISO/IEC, 1994), the inter-frames are predicted from an intra or another inter-frame with a motion estimation (ME) and a motion compensation (MC). Some artefacts may appear in the logo region in the semi-transparent case and in the neighborhood due to these movements in the pictures. (Chang et al., 1995) re-estimate the motion for the macroblocks along the border of the foreground media to be composited. The new prediction error (PE) and the motion vector (MV) are calculated in a motion compensated DCT domain. The lower complexity of the motion compensation in the DCT domain (MC-DCT) has been verified by (Merhav et al., 1996), due to the sparseness of the DCT information. The issue of this temporal prediction is treated in the second section. Our analytic method does not destroy motion information but takes into account a local MC-DCT of the background which is stored in a small memory corresponding to the block-based extended logo region. We close the paper discussing the compromise between the results and decoding degree of the bitstream.

4.3 Exploitation of the DCT Properties

4.3.1 Opaque overlapping

To begin, we consider the simplest case: intra-frames. The data in the bit stream is composed of the variable length codes (VLC) of the quantized DCT coefficients of a 8x8-pixel

block B_{vid} for an intra-frame. The opaque composition consists of finding and replacing the code of B_{vid} with the B_{logo} code in the bit stream.

4.3.2 Linearity

In the spatial domain, the block B_{trans} that results from a semi-transparent compositing can be seen as a simple linear combination of B_{vid} and B_{logo} with an opacity coefficient α as in (4.1). (4.2) shows how the linearity of the DCT allows this composition in the DCT domain.

$$B_{trans} = (1-\alpha).B_{vid} + \alpha.B_{logo} \tag{4.1}$$

$$DCT(B_{trans}) = (1-\alpha).DCT(B_{vid}) + \alpha.DCT(B_{logo}) \tag{4.2}$$

4.3.3 Orthogonal property

The main difficulty for the DCT domain manipulation is the block structure. Indeed, if the position of the logo in the frame and its dimensions are not multiples of 8, the DCT structure of the video is not aligned with the logo structure. (Chang et al., 1995) propose a tool based on a double matrix multiplication, as shown by (4.3). A rectangular portion of a foreground block B can be extracted and moved to anyplace in a background block B'. Figure 4-1 illustrates the design of the first matrix which extracts a H-pixel-high stripe and translates it from the bottom to the top. Then, the second matrix processes a L-pixel-width stripe from the right to the left. H and L are derived from the logo position. The orthogonal property of the DCT helps the operation to be transposed in DCT domain, as shown by (4.4). Using an inner product between the matrix B and an opacity map also would be efficient for the extraction, but not for the translation.

$$B' = H_{vert} \times B \times H_{horiz} \tag{4.3}$$

$$DCT(B') = DCT(H_{vert}) \times DCT(B) \times DCT(H_{horiz}) \tag{4.4}$$

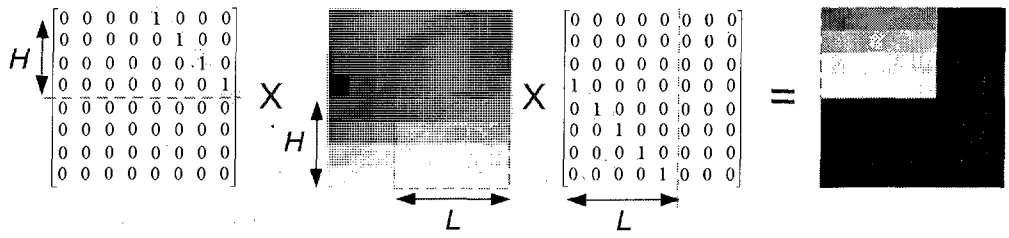


Figure 4-1 Spatial extraction and translation of a *H*x*L* block by the double matrix multiplication.

4.3.4 Implementation for the DCT structure alignment

Our logo source file is a non-coded bitmap so it is processed before the execution of the algorithm. An empty border is added around the logo, as illustrated by the Figure 4-2, before the DCT transform is performed. The border dimensions are derived from the logo position to match with the macroblock structure of the coded video structure. All the tools are available for a position/opacity adjustable compositing in an intra-frame. Figure 4-3 illustrates how the linear combination of three contributions of the background and the logo portion form the block in the top-left corner of the logo. This approach is different because the extraction tool is now applied on a background block only.

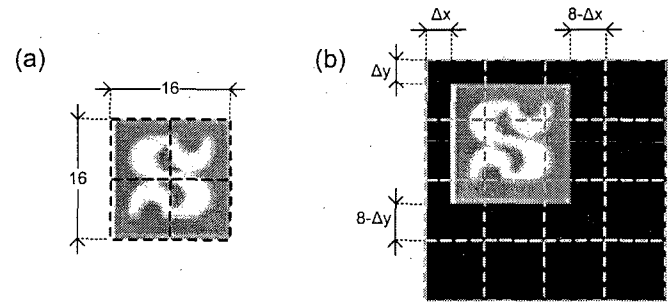


Figure 4-2 Transformation of the source logo (a) adding border (b) before the DCT.

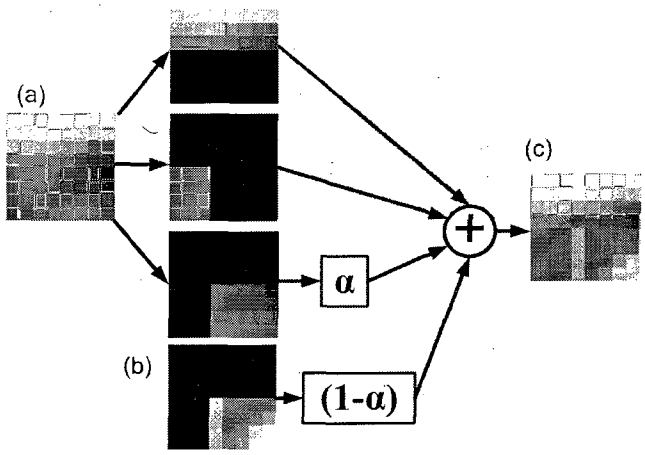


Figure 4-3 The three contributions from the video block (a) and the adjusted logo block (b) are combined to form the final block (c) on the top-left corner of the logo.

4.4 Inter-Frame Processing

4.4.1 Prediction Issues

The following considerations concern the prediction-frames to simplify the explanation, but the principle can be extended to the bidirectional-frames. For a macroblock in a prediction-frame, a ME is performed from a previous (intra/prediction) frame and gives a MV and an associated PE, as described in (ISO/IEC, 1994). The temporal prediction induces mistakes because the first reference intra-frame has been modified due to the compositing of the logo. Therefore, we can adjust both the MV and the DCT of the PE to perform the correction.

The logo is assumed to be constant in pixel values and in position. Its blocks and the MVs associated to its macroblocks are simply reset for an opaque overlapping. However, the MVs can not be fixed to zero in the semi-transparent case because the original background will be distorted. So, the PE should be corrected to avoid a moving logo because the prediction will be affected by the logo data in the decoder. The neighbourhood of the logo needs a special care because the MC may give a macroblock containing a portion of the logo but the reconstructed macroblock should contain only original video data. This is the background uncovering problem. The inverse problem (covering) may occur in the logo zone when MC gives an original prediction with video data only.

4.4.2 Proposition

Both (Chang et al., 1995) and (Roma et al., 2002) propose a transcoder structure. The first need a MC-DCT and new ME-DCT after the compositing and the second authors insert the composition between two MC-DCT. However, this paper focuses on the PE correction. This adjustment is derived from the analysis of the modified prediction in the decoder. Let us denote $\vec{v}$ the MV, $B_{logo}^{\vec{0}}$ the current logo block, $B_{logo}^{\vec{v}}$ its motion compensated version, $P_{vid}^{\vec{v}}$ the original prediction block and E_{vid} the original PE. Figure 4-4 illustrates the worst situation because both the current inter-block to be reconstructed and the prediction referred by $\vec{v}$ in the MC are located on the edge of the logo. In the region n°1, the reconstructed block needs $B_{logo}^{\vec{0}}$, however, the prediction is no longer $P_{vid}^{\vec{v}}$ but now contains $B_{logo}^{\vec{v}}$. Then, $-\vec{v}$ has a "zone entering" vertical component and a "zone exiting" horizontal component, respectively corresponding to the covering (region n°2) and uncovering issues (region n°3). The content of the needed reconstructed block and the prediction must be analysed in these regions to perform the correction. Finally, there is no problem for the region n°4 because both the reconstructed data and the prediction won't have logo information during the decoding process. It results in the correction of the E_{vid} depending on four segmented regions.

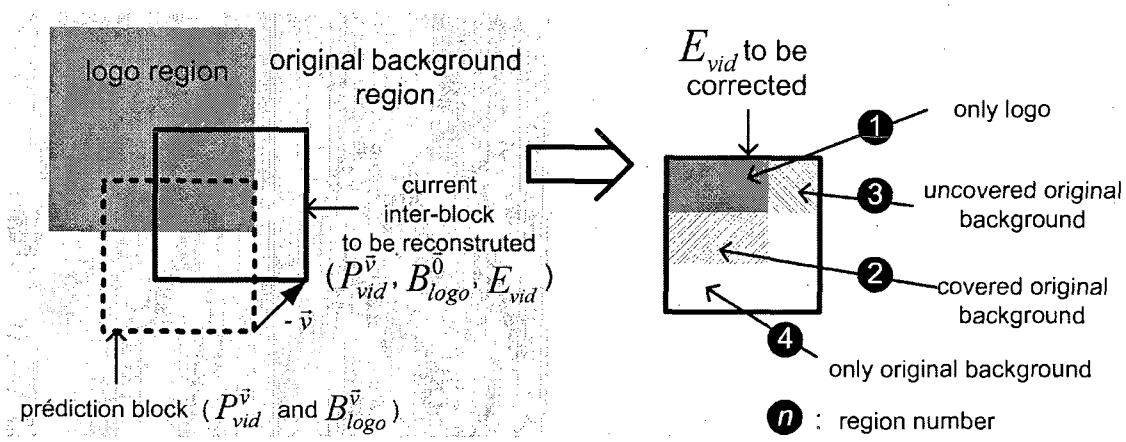


Figure 4-4 Worst case for the prediction issue: E_{vid} must be segmentated in four regions

Table 4-1 details each block (prediction and decoded) for each region and reports the modifications for the PE. All corrections can be processed in the DCT domain using the linearity properties of the DCT transform. The assembling of the four blocks is allowed in this domain by the orthogonal property, in the same way as in Figure 4-3.

TABLE 4-1 FOUR TYPES OF MODIFICATION FOR PREDICTION ERROR

Region	Data type	Needed reconstructed block	Prediction in the decoder	Modified prediction error
❶	logo & background	$(1-\alpha)\left(P_{vid}^{\vec{v}}+E_{vid}\right)+\alpha B_{logo}^{\vec{0}}$	$(1-\alpha)P_{vid}^{\vec{v}}+\alpha B_{logo}^{\vec{v}}$	$(1-\alpha)E_{vid}+\alpha\left(B_{logo}^{\vec{0}}-B_{logo}^{\vec{v}}\right)$
❷	logo & covered background		$P_{vid}^{\vec{v}}$	$(1-\alpha)E_{vid}+\alpha\left(B_{logo}^{\vec{0}}-P_{vid}^{\vec{v}}\right)$
❸	uncovered background	$P_{vid}^{\vec{v}}+E_{vid}$	$(1-\alpha)P_{vid}^{\vec{v}}+\alpha B_{logo}^{\vec{v}}$	$E_{vid}+\alpha\left(P_{vid}^{\vec{v}}-B_{logo}^{\vec{v}}\right)$
❹	original background		$P_{vid}^{\vec{v}}$	E_{vid}

$B_{logo}^{\vec{0}}$ and $B_{logo}^{\vec{v}}$ are easily accessible from the source bitmap. E_{vid} is extracted from the bitstream. The only constraint comes from $P_{vid}^{\vec{v}}$ which is needed to resolve the covering/uncovering issue. So, we feature the MC of the original background in the DCT domain as described in (Merhav et al,. 1996). A reference block pointed at by $\vec{v}$ is generally located on four DCT blocks. The algorithm extracts, translates and sums the contribution of each to form $DCT(P_{vid}^{\vec{v}})$. It's preferable to limit the MC only to the logo region and to a 1-macroblock large border around it in order to restrain the amount of decoded video data as much as possible. This method supplies a useful memory of the moving background which is destroyed in (Chang et al., 1995).

4.5 Implementation and Results

4.5.1 Technical Considerations

Let us give some technical details of our application to match with the MPEG-2 algorithm implementation before presenting the results. The first consideration deals with the intra-frames. The DC coefficient of a block of each of the color components are coded with a Differential Pulse Code Modulation (DPCM) from the previous block in the bitstream. Therefore, the modifications in the logo zone induce an error that propagates until the next macroblock after the logo zone has its 3 DC coefficients corrected. The MVs are also DPCM coded so the modification of one of these should be followed by the same readjustment.

At the present time, our application works only for the non-scalable profiles of MPEG-2 and for progressive intra/prediction frames. But it manages the half pixel precision of the MVs, as required by the standard. It is also able to adapt to the 3 chroma formats, even if the main broadcasting format is format 4:2:2 (the chroma is horizontally sub-sampled).

4.5.2 Results

The algorithm is tested on a 32x32- pixel logo and two coded SIF format (352x240 pixels, 4:2:2) sequences with a 4 Mbits/s bit rate. The first sequence is composed of the twelve first pictures of "Tennis". The coding type sequence is *I,P,P,I,P,P,I...* and $\alpha = 50\%$. The second sequence is based on "Flowers" and its GOP structure is *I,P,P,P,I,P,P,P,I...* and α is fixed to 100%. In order to get an objective comparison, we decode the original bitstream and overlap the logo in the spatial domain, like in the traditional process.

Figure 4-5 illustrates how very similar the two sequences are visually compared to their references. The zone is restrained to 96x96 pixels for a better visualization and for a more substantial PSNR calculation. Figure 4-6 shows the PSNR evolution and confirms the quality of the images. For each sequence, PSNR decreases of about 3dB with α from 20% to 100%. The more significant the logo is, the more different from the non-quantized original used in the reference spatial compositing it will be. Indeed, the reference images contain a most significant non-coded portion when $\alpha = 100\%$.

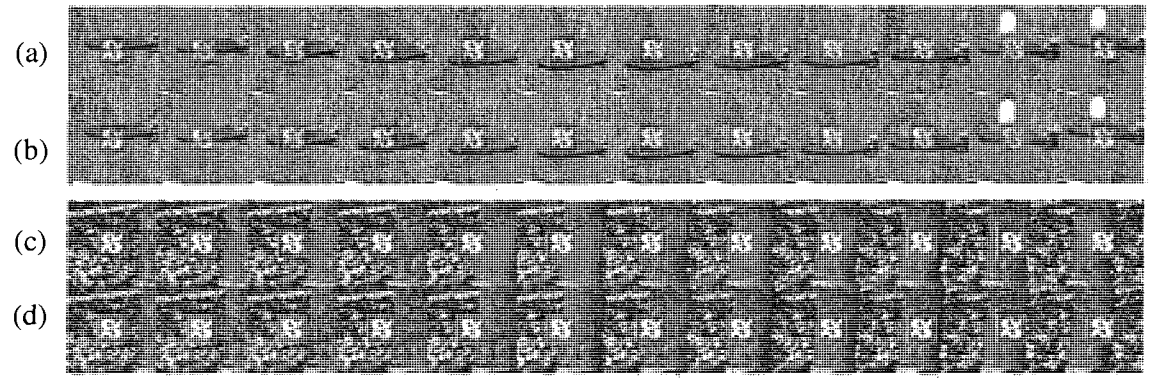


Figure 4-5 Sequences: (a) processed tennis, (b) reference tennis, (c) processed flower garden, (d) reference flower garden

We keep in mind that these enthusiastic results depend on our initial constraints. The bitstream decoding degree is the main factor of performance. The logo transparency and the block segmentation in a transform domain require the linear and the orthogonal properties of the transform. However, the VLC, the quantization and the motion compensation do not present these characteristics. The algorithm is based on an analytic solution and needs a memory of the original background to solve the covering/uncovering trouble. But this zone stays relatively small when comparing to the picture size.

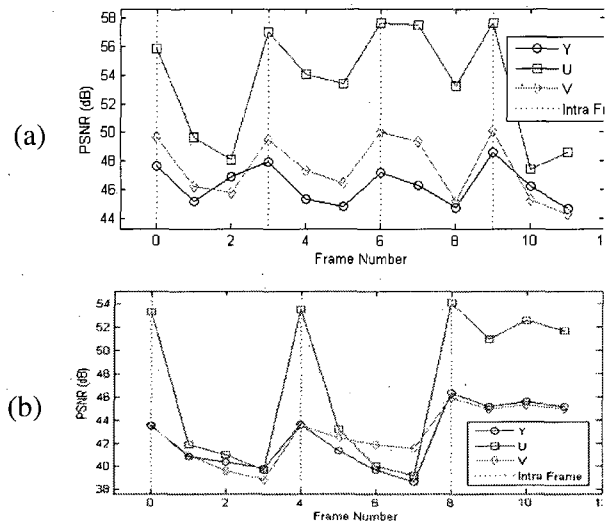


Figure 4-6 PSNR evolution for the sequence "Tennis" (a) and "Flower Garden" (b).

4.6 Conclusion

This paper proposes a concrete application of DCT domain operating tools for image processing. The complexity is decreased thanks to the many zero DCT coefficients. We achieve a good adjustable logo compositing in a MPEG-2 bitstream which contains intra/prediction frames. This analytic method could be a reference for further studies that tend toward a more empirical solution or a practical engineering. However, this near optimal solution could find easy extensions like the bidirectional frame and the interlaced coding features. Also, it is possible to create a real-time management of the logo position and opacity with a back up system to prevent prediction issues.

5 PRÉSENTATION DES RÉSULTATS

Comme cette publication précédente ne donne pas des résultats de façon exhaustive, ce dernier chapitre vient compléter cette étude et met mieux en évidence l'influence de paramètres comme le coefficient d'opacité, les dimensions du logo, etc. C'est également l'occasion de présenter l'environnement de développement et de test mise en œuvre pour les besoin du projet.

5.1 Protocole expérimental

Un schéma d'ensemble de l'environnement de développement est présent à l'Annexe A. L'ensemble de l'algorithme a été codé en langage *C*. Certaines parties ont été directement prélevées dans le codeur et le décodeur de l'implémentation logicielle *MSSG-TM5* (MPEG Group, 1996); c'est le cas pour les modules de codage entropique, de quantification et de lecture des entêtes de la structure de donnée du flux binaire. Les fichiers *MPEG-2* ne sont pas directement accessibles ; ils sont générés par un codeur qui prend en entrée des séquences de test standard fournies par *The Center for Image Processing Research* [26]. On contrôle précisément de cette façon le contenu du flux binaire. Un décodeur est employé pour obtenir à la fois les images du nouveau fichier généré par le module de composition *DCT*. L'Annexe B présente ce module comme la synthèse de l'algorithme. Le fichier *MPEG-2* original passe lui aussi au travers du décodeur pour retrouver les images d'origines reconstruites. On applique alors sur ces dernières l'incrustation du logo, à la manière traditionnelle, c'est-à-dire dans le domaine spatial. On obtient alors une bonne référence pour la comparaison subjective et objective car si on n'avait pris les images d'origines comme référence, il y aurait eu de légères erreurs de quantification à chaque pixel. Finalement, on fenêtre les images en laissant une bande 16 pixels tout autour dans le but d'obtenir une visualisation du résultat plus adéquate et aussi de permettre le calcul d'un *PSNR* (*Peak Signal to Noise Ratio*) plus

constant. Le tout est supervisé par un script *Matlab* qui génère également une interface graphique pour l'utilisateur/développeur ; celle-ci est visible à l'Annexe C.

5.2 Résultats

Certains éléments ont déjà été mentionnés dans l'article mais il convient d'y apporter quelques compléments. En effet, on peut légitimement s'intéresser plus en détail à la question de savoir si les dimensions, l'opacité et la position du logo ont une influence sur la qualité du résultat.

5.2.1 Signaux test

TABLEAU 5-1 DESCRIPTION DES SÉQUENCES ET DES CONDITIONS DE TESTS

Test	1	2	3
Nom de la séquence	*« Tennis »*	*« Flower garden »*	*« Carphone »*
Dimensions	352 x 240	352 x 240	176 x 144
Format de la Chrominance	4 : 2 : 0	4 : 2 : 2	4 : 2 : 0
Structure *GOP*	*I,P,P,I...*	*I,P,P,P,I...*	*I,P,P,P,P,P,I...*
Opacité du Logo	50%	60%	30%
Mouvements de l'arrière plan	progressifs	unidirectionnels	saccadés, multidirectionnels
Textures de l'arrière plan	peu	beaucoup	mixte

Le logo, quant à lui, est prélevé sur le site de l'Université de Sherbrooke. Ses dimensions sont 32x32 pixels (4x4 blocs ou 2x2 macroblocs) pour s'adapter à la structure macrobloc de l'arrière plan lorsque que les composantes de sa position sont multiples de 16.

5.2.2 Séquences

Les figures 5.1 à 5.3 montrent les séquences traitées (au-dessus) comparées aux séquences de référence (en-dessous). Dans chaque cas, le PSNR est calculé dans la fenêtre d'affichage et pour chacune des composantes de couleur dans l'espace YUV. Première constatation, on ne voit quasiment pas à l'œil nu la différence entre les images modifiées dans le domaine DCT de celles modifiés dans le domaine spatial. Effectivement, le *PSNR* vient confirmer la qualité des résultats car il ne passe jamais en dessous des 35dB. Celui de la luminance est globalement inférieur car il contient de 2 à 4 fois plus coefficients modifiés que la chrominance. Le raisonnement peut être étendu pour expliquer que la composante *U* est toujours mieux traitée car elle contient beaucoup plus de coefficients nuls, donc non modifiés. Ensuite, un pic est visible à chaque début de *GOP* car il s'agit d'images *intra*. Ces dernières ne nécessitent pas les opérations supplémentaires pour corriger l'erreur de prédiction en fonction de l'arrière plan qui bouge. De manière générale, plus il y a d'opérations, plus il y a d'imprécision à cause des arrondis dans le domaine *DCT*.

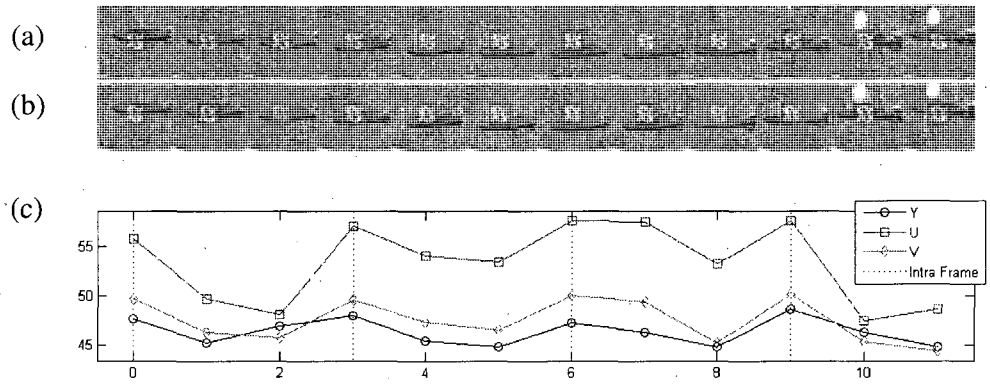
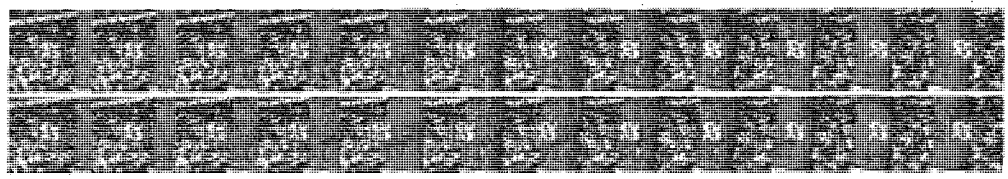


Figure 5-1 Séquence «*Tennis* » traitée (a), celle de référence (b) et mesure du *PSNR* (c), avec une opacité de 50%.

(a)

(b)



(c)

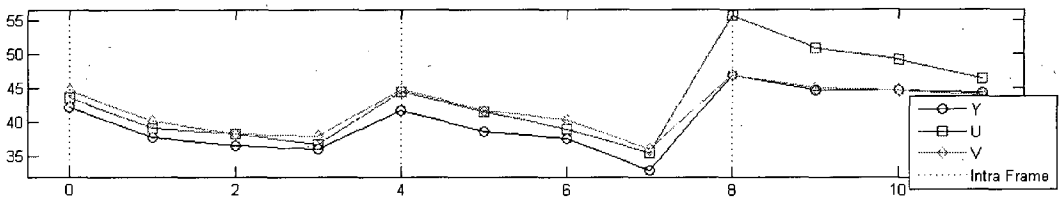
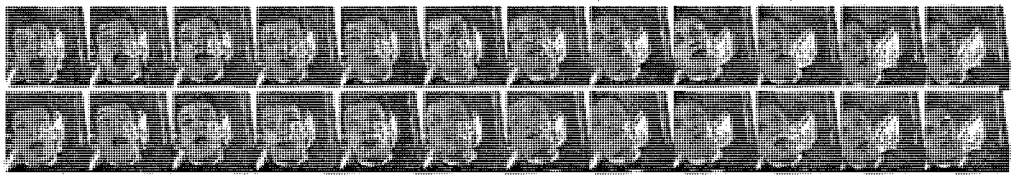


Figure 5-2 Séquence «*Flower Garden* » traitée (a), celle de référence (b) et mesure du *PSNR* (c), avec une opacité de 60%

(a)

(b)



(c)

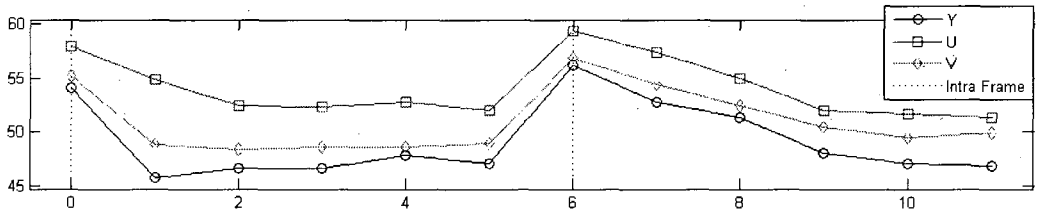


Figure 5-3 Séquence «*Carphone* » traitée (a), celle de référence (b) et mesure du *PSNR* (c), avec une opacité de 40%.

5.2.3 Influence de l'opacité α

Cette fois-ci, le critère objectif est la moyenne du *PSNR* de la luminance mesure pour chaque même séquence. La Figure 5-4 indique son évolution si on fait varier α par pas de 20%. On note une légère diminution de quelques dB, ce qui n'est pas très significatif. Plus l'opacité est

importante, plus la part d'information du logo dans le bloc de référence non codé, donc non quantifié, diffère de celle dans le bloc *DCT* quantifié.

En revanche, le *PSNR* remonte légèrement lorsque la superposition devient complètement opaque. Cela s'explique par le traitement spécial de l'algorithme qui annule toute prédiction dans ce cas particulier, ce qui signifie que les blocs *inter* sont rigoureusement les mêmes que les blocs *intra* dans la première image de référence du *GOP*. Si, en plus de cette condition, la structure du logo coïncide avec celle de l'arrière plan, alors la tâche est d'autant plus simplifiée puisqu'il n'y même plus besoin de réaliser la quantification inverse des blocs *DCT*.

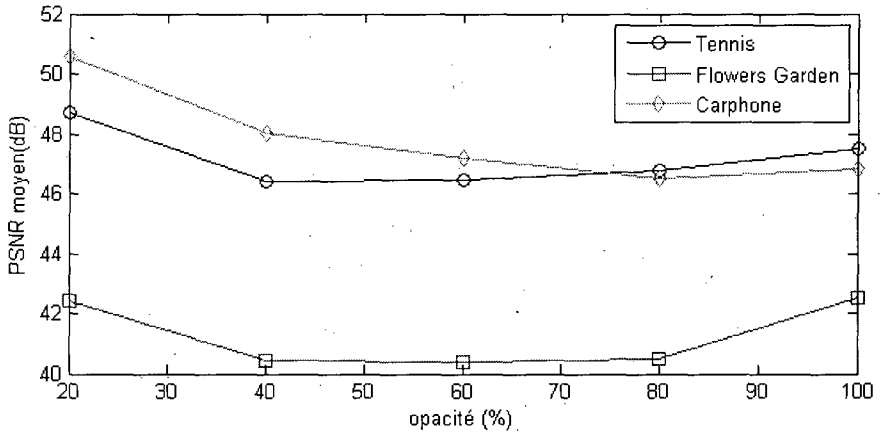


Figure 5-4 Évolution du *PSNR* moyen de la luminance en fonction de l'opacité α du logo.

5.2.4 Influence de la position du logo

Dans la suite de l'étude, on vérifie si le fait de varier légèrement la position du logo influence beaucoup le résultat. En effet, si les coordonnées de la position (Δx,Δy) sont toutes les deux non multiples de 8, un logo de N x M blocs de luminance chevauche $(N+1)$x$(M+1)$ blocs de l'arrière plan. Il y a donc $N+M-1$ blocs supplémentaires qui exigent une correction entraînant des arrondis et des erreurs de quantification. Le raisonnement peut être étendu au niveau macrobloc (avec (Δx, Δy) multiples de 16 ou non) car la chrominance est sous-échantillonnée pour les formats 4:2:2 et 4:2:0. Pour avoir une idée des conséquences, différentes images erreur *intra* sont formées à la Figure 5-5 et le *PSNR* moyen de la luminance est calculé sur

l'ensemble de la séquence « *Tennis* », avec le même logo de 32x32 pixels (4x4 blocs) et α=60%. Il est clairement visible que l'influence reste mineure ; c'est d'autant plus vrai si le logo est grand.

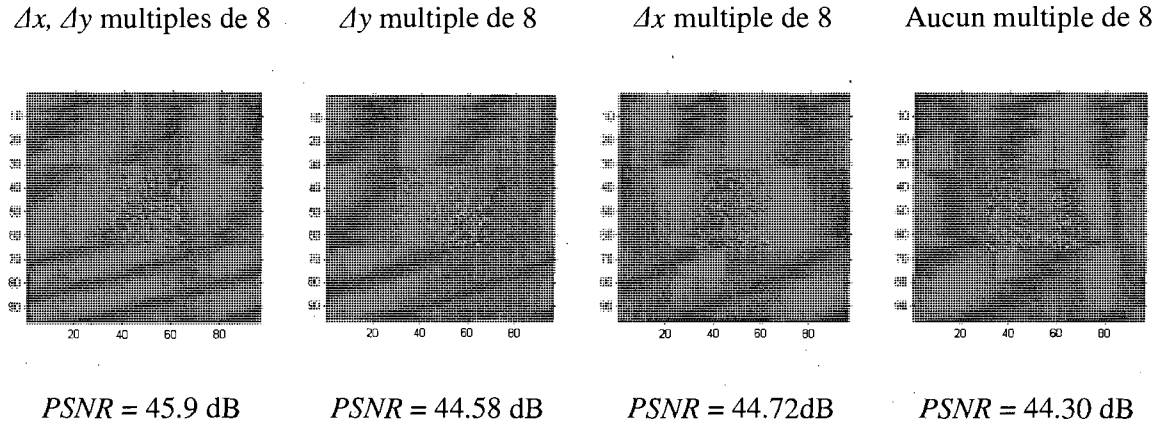


Figure 5-5 Influence de la position du logo sur le *PSNR* de la luminance.

5.2.5 Étude de la taille du logo

La question de la taille du logo est justifiée par rapport au format de l'image. Les formats *QCIF* (176x144) et *HD* (1920x1152) en sont deux exemples extrêmes. Alors en admettant que les dimensions sont fixées à 10% de celle de l'image, ceci est équivalent à une plage de 1 à 56 macroblocs. Le tableau 5-2 fournit les résultats sur une plage d'étude légèrement réduite. Une fois encore, le *PSNR* est calculé dans une fenêtre qui entoure le logo avec une marge de 1 macrobloc et il est moyenné sur l'ensemble de la séquence « *Tennis* ». On retrouve le même phénomène que précédemment, à savoir que plus il y a de blocs concernés, plus les coefficients modifiés et arrondis sont nombreux.

TABLEAU 5-2 INFLUENCE DE LA TAILLE DU LOGO

Taille du logo en macroblocs	**4**	**8**	**16**	**32**
***PSNR* moyen (dB)**	45.84	43.20	42.1	39.8

5.3 Commentaires

L'efficacité de l'algorithme est ainsi validée car même si le *PSNR* diminue, le résultat visuel est tout à fait satisfaisant. Les zones de textures et de contour sont, comme souvent en traitement d'images, les plus sensibles aux opérations mais le Système Visuel Humain perd de sa précision à ces endroits. La règle générale, et aussi intuitive, qui se dégage de cette étude est que la façon d'optimiser la performance est de limiter le nombre d'opérations telles que la multiplication qui induit des imprécisions. Pour cette raison, le mieux est d'avoir un logo de taille réduite et de le placer à une position multiple de 16 avec une opacité totale.

La contrainte de limitation du décodage du flux binaire a indirectement été considérée. Le décodage entropique est inévitable car il est impossible de réaliser des opérations sur des symboles binaires. Ensuite, il est nécessaire de réaliser la quantification inverse qui aurait pu être proche d'une opération linéaire si le pas de quantification avait été constant, mais ce n'est pas le cas. Ces deux étapes ne posent pas de problème car elles sont beaucoup moins gourmandes en ressource que la *DCT*. Enfin, une certaine forme de compensation de mouvement fournit, sous forme *DCT*, la prédiction de la vidéo originale nécessaire à la correction du recouvrement et du découvrement. Toutes ces opérations pour atteindre le domaine *DCT* et les degrés de liberté qu'il procure peuvent être évités seulement si l'on dispose de conditions très particulières pour le logo (position multiple de 16, opacité totale) mais également pour la vidéo qui devrait contenir aucun ou très peu de mouvement pour éviter le découvrement.

Par contre, cette étude n'a pas la vocation de prouver le bien fondé d'opérer dans le domaine *DCT* plutôt que dans le domaine spatial, puisque c'est une de nos hypothèses. Il aurait fallu pour cela quantifier la complexité de calcul du processus entier et le comparer avec la transformée *IDCT* suivie de la composition dans le domaine spatial et de la *DCT*. De telles mesures impliquent une étude statistique selon taux de coefficients *AC* nuls, lui-même dépendant du contenu de l'image, des mouvements et du débit imposé.

CONCLUSION

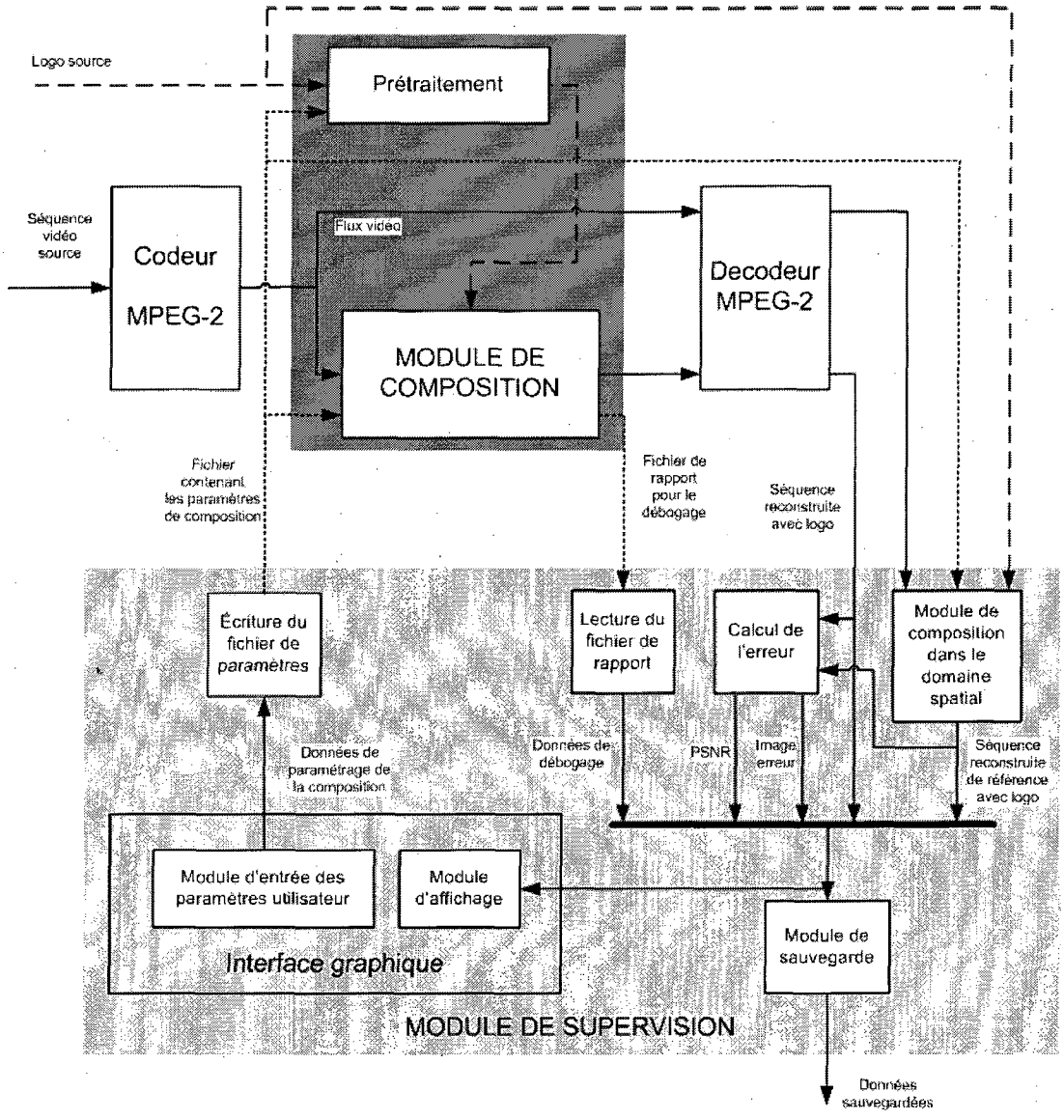
L'algorithme développé pour l'incrustation d'un logo dans un fichier *MPEG-2* est plus qu'une simple application des principes établis par les travaux de Chang et Messerschmitt (Chang et al., 1995). La méthode mise en œuvre s'en démarque pour gérer les mouvements de l'arrière plan, d'une part, du fait que ces auteurs traitent quasiment pas de la semi-transparence du premier plan. Mais surtout, l'utilisation d'une version *DCT* de l'image originale reconstruite évite de détruire l'information de mouvement lors de son recouvrement par le logo et d'avoir à la recréer lors du découvrement. De plus, il s'agit d'une solution analytique et le dernier chapitre valide sa fonctionnalité par une mesure du *PSNR* et par un résultat visuel tout à fait satisfaisant.

La motivation de ce genre d'étude est de limiter au maximum le décodage du flux binaire mais le cahier des charges a imposé quelques compromis. L'opacité ajustable et le positionnement du logo au pixel près semble impossible sur des données brutes. Ce travail s'inscrit dans la lignée de ceux qui se focalisent sur le traitement d'images dans le domaine *DCT* car celui-ci accorde une bonne liberté de manœuvre. Et comme son information est clairsemée, les opérations dans ce domaine sont déjà vraiment plus rapides que dans le domaine spatial car une grosse quantité de coefficients *DCT* nuls diminue considérablement le nombre de multiplications.

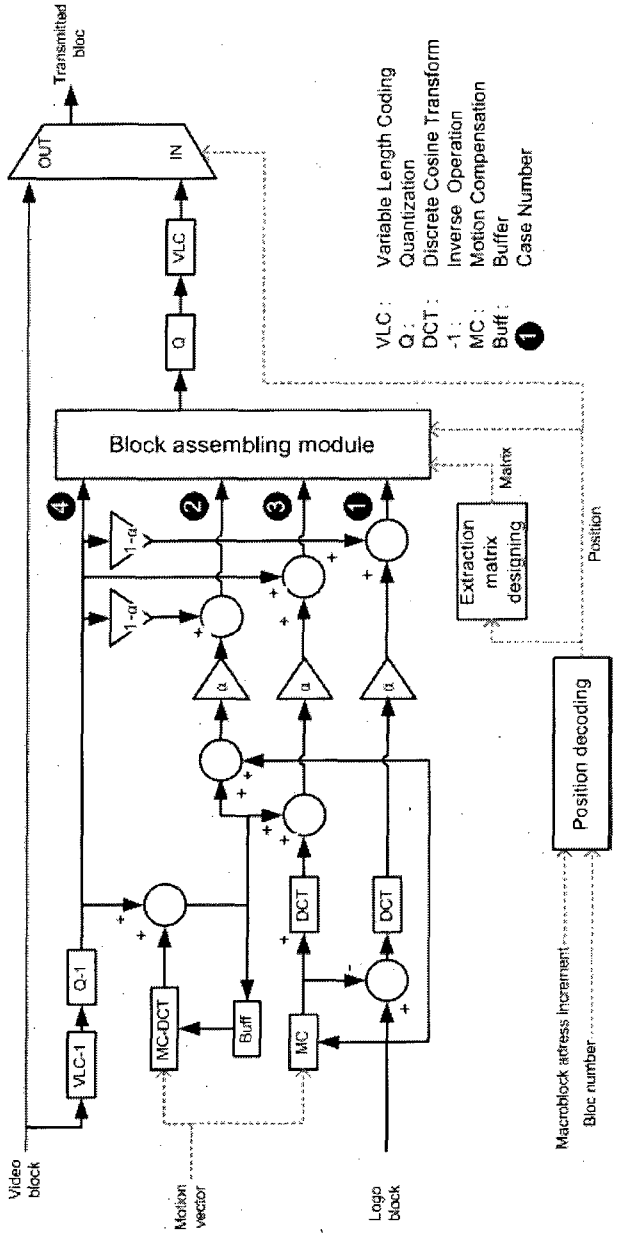
Ce travail pourrait servir de référence à d'autres études qui voudraient s'écarter de la solution analytique pour limiter encore plus le décodage (par exemple s'affranchir de la compensation de mouvement dans le domaine *DCT*). Cependant, des extensions de ce projet sont faciles à imaginer à court terme. La prise en charge du codage en mode entrelacé, en plus du mode progressif, et des modes de fonctionnement échelonnables prévus par les spécifications du codec en sont deux exemples. On peut également imaginer insérer un plan intermédiaire entre l'arrière et le premier plan. Une incrustation dynamique est tout aussi envisageable en faisant

varier l'opacité, la position, la forme ou la couleur du logo. La vocation d'un tel dispositif pourrait aboutir au mixage de deux sources codées MPEG-2. Ceci est déjà réalisé pour l'affichage multi-canal (exemple : la vidéo conférence, TV par satellite). Leurs contenus se juxtaposent mais ne se chevauchent pas encore. Gérer les conflits de mouvements entre les deux sources apporterait une contribution considérable.

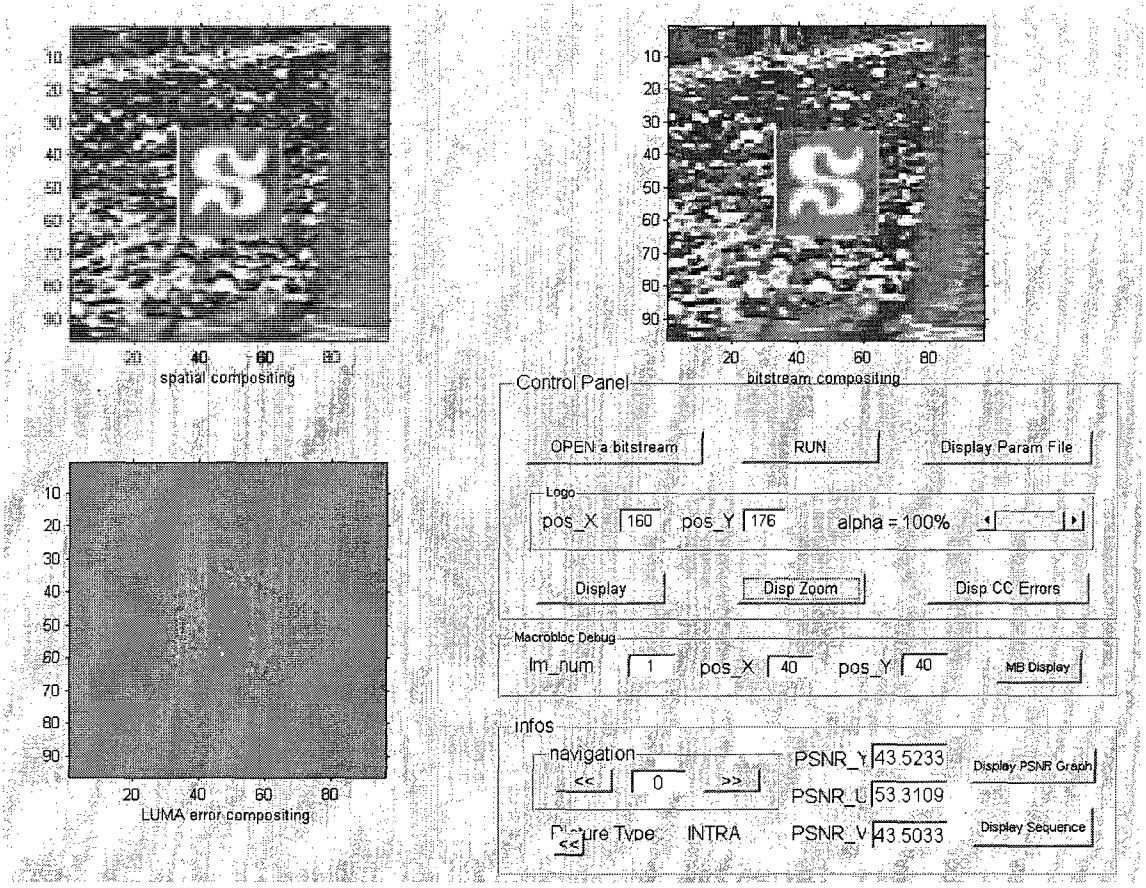
Annexe A Environnement de développement



Annexe B Schéma fonctionnel du module de composition



Annexe C Interface graphique



Bibliographie

[1] Cho N. I. and Lee S. U. (1991) "Fast algorithm and implementation of 2-D discrete cosine transform", *IEEE Transactions on Circuits and Systems*, vol. 38, pp. 297–305.

[2] Chitprasert B. and Rao K. R. (1990). "Discrete cosine transform filtering", *IEEE Transactions on Signal Processing*, vol. 19, pp. 233–245.

[3] Martucci S. A.(1994). "Symmetric convolution and the discrete sine and cosine transform", *IEEE Transactions on Signal Processing*, vol. 42, pp. 1038–1051.

[4] Merhav N. and Bhaskaran V. 1995 "A fast algorithm for DCT domain filtering", *HPL Technical Report*, HPL-95-56, Hewlett-Packard Labs.

[5] Kresch R. and Merhav N. (1999). "Fast DCT domain filtering using the DCT and the DST", *IEEE Transactions on Image Processing*, vol. 8, pp. 821–833.

[6] Changhoon Y. (2004). "An efficient method for DCT-domain separable symmetric 2-D linear filtering", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 4.

[7] Triantafyllidis G. A., Tzovaras D., and Gerassimos Strintzis M. (2002). "Blocking artifact detection and reduction in compressed data", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 10.

[8] Chen T., Wu H. R. and Qiu B. (2001). "Adaptive post filtering of transform coefficients for reduction of blocking" *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 5.

[9] Paek H., Kim R.-C., and Lee S.-U. (1998). "On the POCS-based post processing technique to reduce the blocking artifacts in transform coded images", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, pp. 358–367.

[10] Park H. W., Park Y. S., and Oh S.-K. (2003). "L/M-fold image resizing in block-dct domain using symmetric convolution", *IEEE Transactions on Image Processing*, vol. 12, no. 9.

[11] Fung K.-T. and Siu W.-C. (2006). "DCT-based video downscaling transcoder using split and merge technique", *IEEE Transactions on Image Processing*, vol. 15, no. 2.

[12] Merhav N. and Bhaskaran V. (1997). "Fast algorithms for DCT-domain image down sampling and for inverse motion compensation", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, pp. 468–476.

[13] Goto T., Shinkai Y. and Sakurai M. (2008). "Resolution conversion method for rational scale using neighboring blocks' DCT Coefficient", in *Congress on Image and Signal Processing*, pp. 513-517, Sanya, Hainan, China, May 2008.

[14] Martins B. and Forchhammer S. (2002). "A unified approach to restoration, deinterlacing and resolution enhancement in decoding MPEG-2 video", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 9.

[15] Ghanbari M. and Assunçao P. A. A. (1997). "Transcoding of MPEG-2 video in the frequency domain", in *Proceedings ICASSP*, vol. IV, pp. 2633–2635.

[16] Fung K.T., Chan Y.-L., and Siu W.-C. (2004). "Low-complexity and high-quality frame-skipping transcoder for continuous presence multipoint video conferencing", *IEEE Transactions on Multimedia*, vol. 6, no. 1, pp. 31-46, .

[17] Ishwar B. S. and Sethi K. (1998). "Block-based manipulations on transform-compressed images and videos", *Multimedia Systems*, vol. 6, pp. 113–124.

[18] Jòsson R. (1997). "Efficient DCT domain implementation of picture masking and compositing", in *International Conference on Image Processing*, vol. 2, pp. 366-369, Santa Barbara, CA.

[19] Chang S.-F. and Messerschmitt D. G. (1995). "Manipulation and compositing of MC-DCT compressed video", *IEEE Journal on Selected Areas Communication*, vol. 13, pp. 1–11.

[20] Merhav N. and Bhaskaran V. (1996). "A fast algorithm for DCT-domain inverse motion compensation", *Journal of Visual Communication and Image Representation*, vol. 7, no. 4, pp. 395-410.

[21] Roma N. and Sousa L., « Insertion of Irregular-Shaped Logos in the compressed domain », The 14th International Conference in Digital Signal Processing, Santorini, Greece, July 2002.

[22] Panusopone K, X. Chen and F. Ling, "Logo Insertion in MPEG Transcoder", Proc. IEEE International Conference on ASSP, Salt Lake City, USA, May 2001.

[23] ISO/IEC (1994). Information Technology-Generic Coding of Moving Pictures and Associated Audio, ISO/IEC 13 818-2/3

[24] MPEG Group. (1996). MSSG MPEG-2 video software encoder, TM5. Available: URL http://www.mpeg.org/MPEG/MSSG

[25] Gall D.L. (1991). MPEG: A video compression standard for multimedia applications. *Communication, ACM* 34(4): pp.47–58.

[26] *Center for Image Processing Research*, resources : http://www.cipr.rpi.edu/resource/sequences/index.html.